MICROPROCESSOR PRODUCTS DATA BOOK

January 1983

$7.00

FAIRCHILD

A Schlumberger Company

January 1983

MICROPROCESSOR PRODUCTS DATA BOOK

Microprocessor Division
3420 Central Expressway
Santa Clara, CA 95051

FAIRCHILD
A Schlumberger Company

The Advance Product Information designation on a Fairchild publication indicates that the product described is not characterized. The specifications presented are based on design goals or preliminary part evaluation and, as they are subject to change, are not guaranteed. Fairchild Microprocessor Division should be contacted for current information on these products.

The information furnished in this publication is believed to be accurate and reliable. However, Fairchild cannot assume responsibility for its use, or for use of any circuitry described, other than circuitry entirely embodied in a Fairchild product. No license is granted or implied under any Fairchild patents or trademarks.

Fairchild reserves the right to make changes in the circuitry or specifications presented in this publication at any time and without notice.

Table of Contents

Table of Contents

Table of Contents

Table of Contents

FAIRCHILD
A Schlumberger Company

Section 1
Introduction

General

A microprocessor is essentially an integrated circuit logic replacement device that performs the functions of the central processing unit (CPU) of a computer system. The overall task of the microprocessor is to receive digital data and store it for later processing, to perform arithmetic and logic operations on the data in accordance with instructions contained in a stored program, and to present the results of these operations to the user through some form of output mechanism.

The program is a definable and non-varying specification for any given application. It normally resides in a read only memory (ROM) or program storage unit (PSU). Variable data that is to be operated upon by the microprocessor is normally stored in a random access memory (RAM) or other transient data storage element.

Although architectural details vary depending upon manufacturer and technology, a typical microprocessor comprises the following functional areas:

1. Instruction decoding to interpret program instructions.
2. An arithmetic and logic unit (ALU) to perform binary addition, subtraction, etc., and Boolean logic operations.
3. Registers to temporarily store frequently manipulated data.
4. Address buffers to provide the next program instruction address.
5. Input/output (I/O) buffers to read information into or write information out of the microprocessor.

Microprocessors are generally used in conjunction with support devices that perform timing, program and transient data memory, I/O signal interface, and other functions. A wide range of configurations is possible with a microprocessor and its related devices; each configuration represents a full microcomputer system.

A single-chip microcomputer incorporates CPU, memory, I/O, control, and other functions into one integrated circuit. Typically, such devices have facilities for enhancement of capabilities by interconnection with external devices.

The Fairchild Microprocessor Division product line encompasses microprocessors and their support devices, single- and multi-chip microcomputers, and systems to emulate and develop hardware and software.

Product Line

The Microprocessor Division product line includes a wide range of devices to meet the specific needs of four broad application areas:

1. 8-bit microprocessors
2. 8-bit single-chip microcomputers
3. 16-bit microprocessors
4. Development aids

Within these areas, the Division offers a blend of innovative, state-of-the-art devices and proven, well-established devices. For example, the members of the F6800 family, and of the F8 family, can be configured to create a variety of 8-bit computer systems that have a wide range of capabilities. Similarly, the F9445 family components can create extremely fast 16-bit computer systems that are exceptionally resistant to harsh environments, and the F16000 family members can be used in configurations that are ideally suited to communications applications. (The F16000 has a 16-bit I/O structure and a 32-bit internal architecture.)

To the user, the Microprocessor Division line represents a single source of cost-effective solutions to the full spectrum of application problems.

Data Book

This data book presents a complete technical description of the Fairchild Microprocessor Division product line. Where devices have been characterized, specific information is presented in the form of data sheets. Information on partially characterized devices, and on devices currently under development, is in the form of advance product information sheets. More complete data can be obtained from the Product Marketing Department.

FAIRCHILD
A Schlumberger Company

Section 2
Ordering and Packaging Information

General

Specific ordering codes, as well as the temperature ranges and package types available, are included in each data sheet of sections 3 through 8.

Temperature Range

The basic temperature ranges typically available are:

C Commercial (0°C to 75°C)
L Automotive (– 40°C to 85°C)
M Military (– 55°C to 125°C)

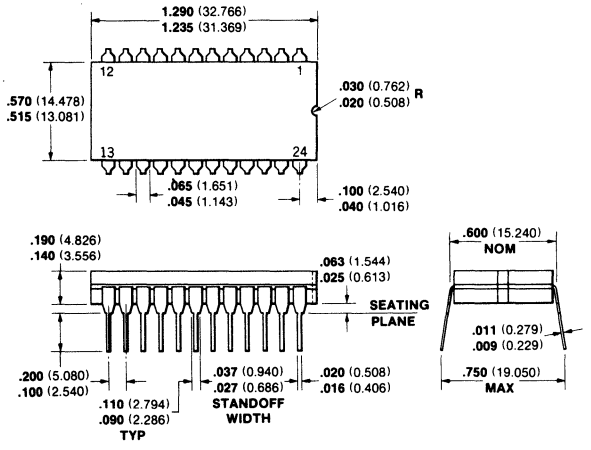
Package Types and Outlines

The basic package type of a device, such as dual-in-line plastic or dual-in-line ceramic, is indicated by the ordering code for that device. To accommodate various die sizes and pin numbers, different package forms exist within each package type.

The package forms indicated by device ordering codes are illustrated in the following detailed outline drawings.

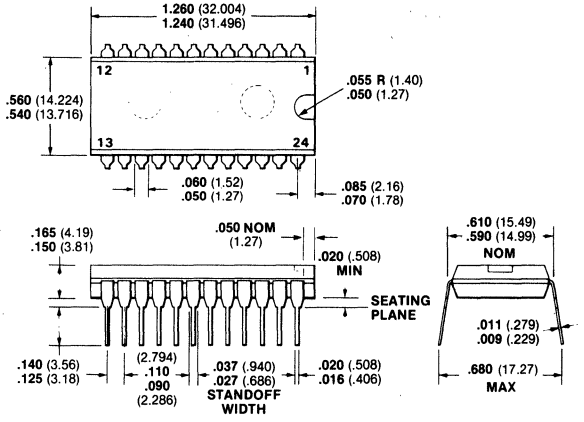
Selected products are optionally available in 44- and 68-pin leadless chip carriers. Contact your local sales office for more information.

24-Pin Ceramic Dual-In-Line



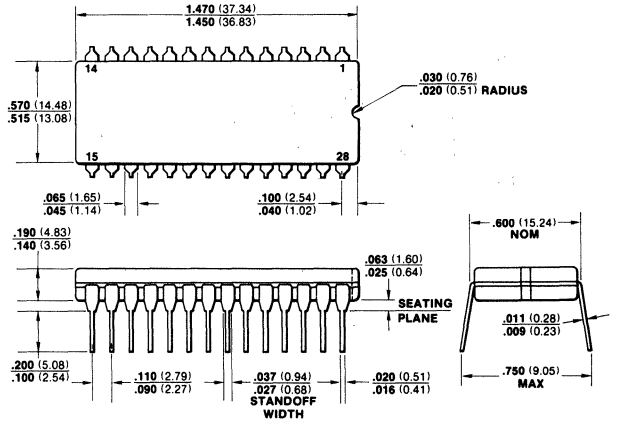
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pin material is nickel gold-plated kovar.
Cap is kovar.
Base is ceramic.
Package weight is 6.5 grams.

24-Pin Plastic DIP



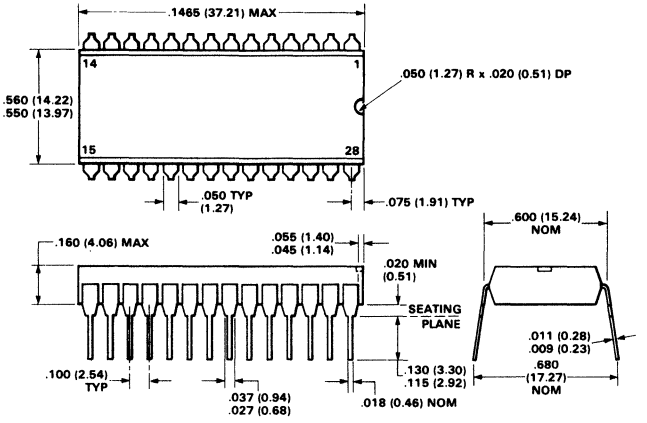
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pins are tin-plated kovar.
Package material is plastic.

28-Pin Ceramic Dual-In-Line



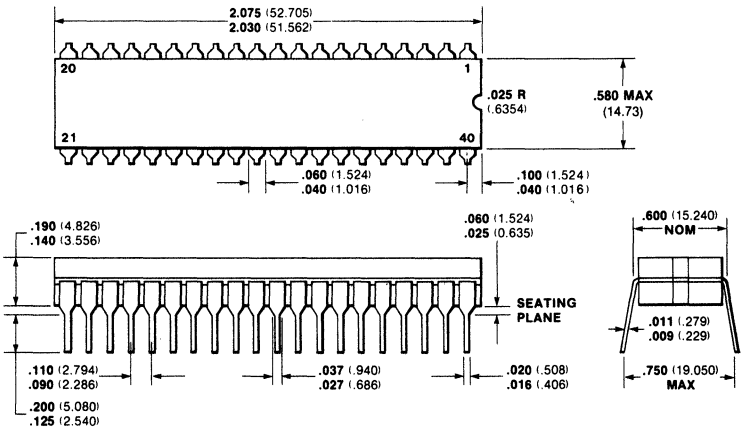
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pin material is nickel gold-plated kovar.
Cap is kovar.
Base is ceramic.
Package weight is 6.5 grams.

28-Pin Plastic Dual-In-Line



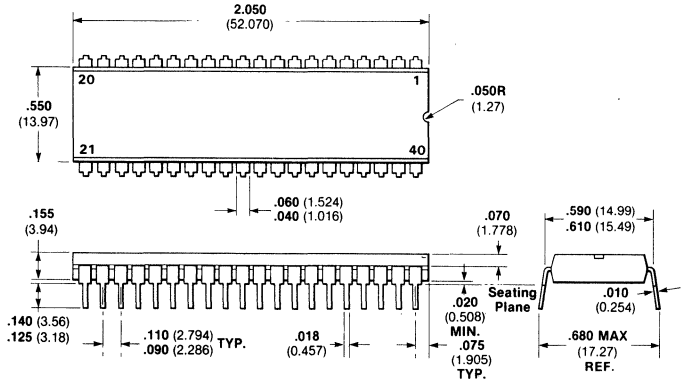
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pins are tin-plated kovar.
Package material is plastic.

40-Pin Ceramic Dual-In-Line



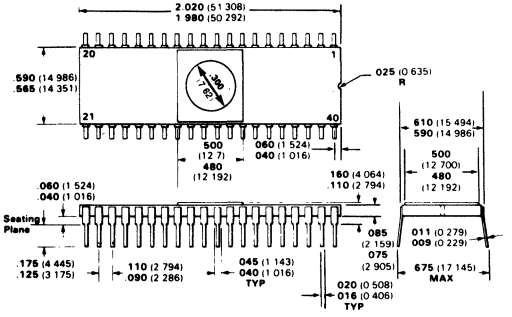
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pin material is nickel gold-plated kovar.
Cap is kovar.
Base is ceramic.
Package weight is 6.5 grams.

40-Pin Plastic Dual-In-Line



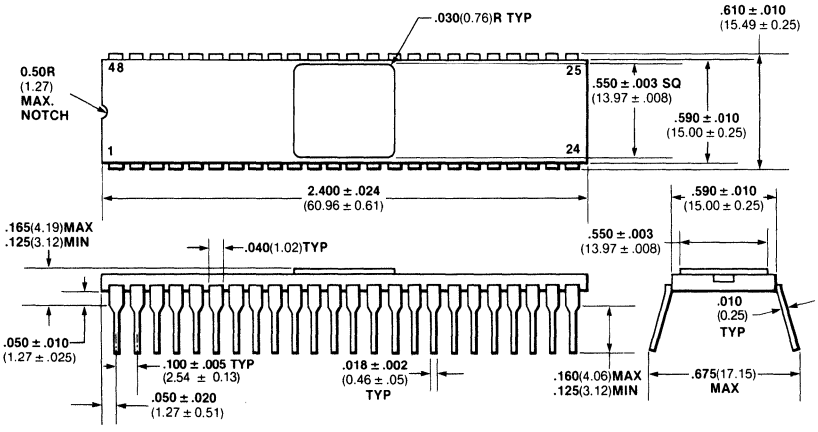
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pins are tin-plated kovar.
Package material is plastic.

40-Pin Ceramic Dual-In-Line (EPROM)



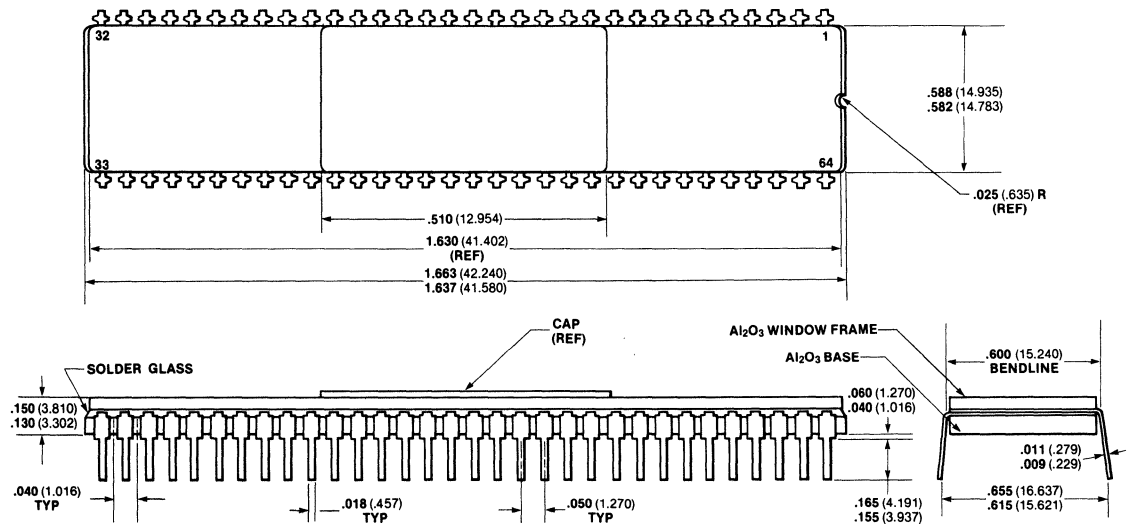
NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pin material is nickel gold-plated kovar.
Cap is kovar.
Base is ceramic.
Package weight is 6.5 grams.

48 Lead Sidebrazed Package



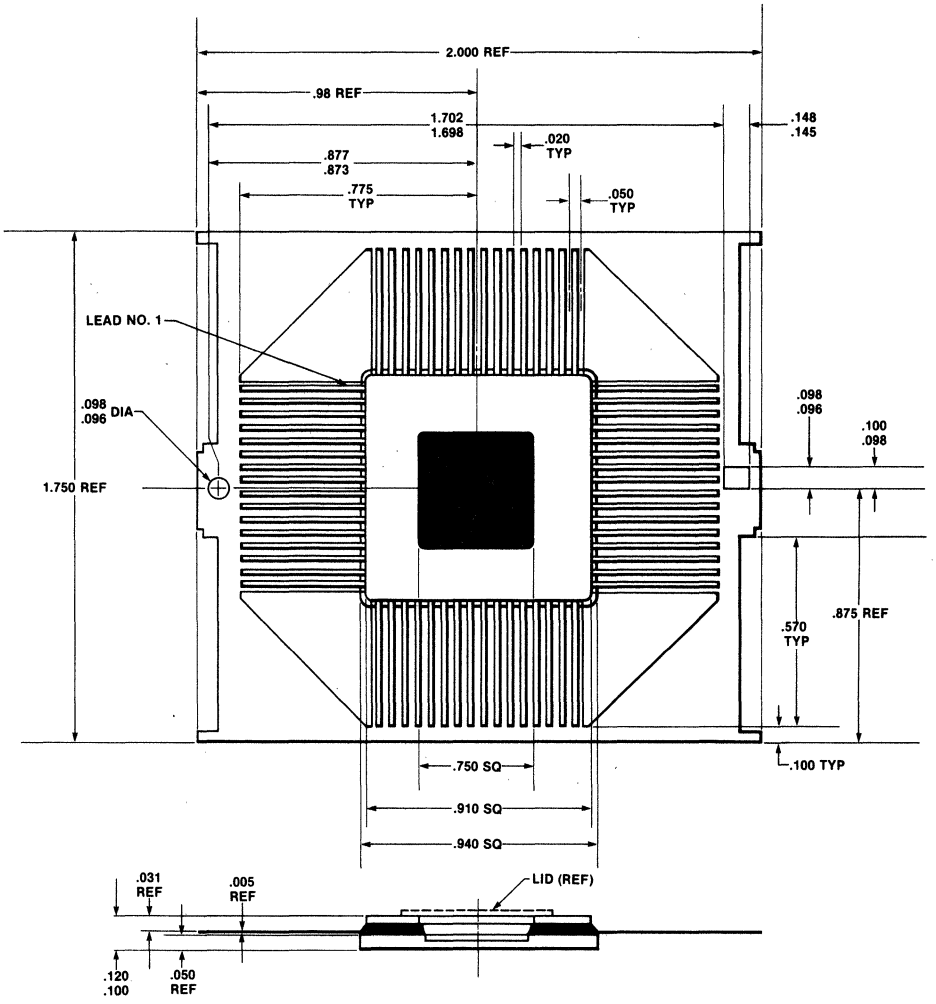
NOTES:
1. Lead Material Is Nickel Gold Plated Kovar or Alloy 42
2. Cap Is Kovar or Alloy 42
3. Base Is Ceramic
4. Cavity Size Is .400 × .400
5. Package Weight Is 7.7 Grams

64-Pin Ceramic Dual-In-Line



NOTES:
All dimensions are in inches **bold** and millimeters (parentheses).
Pin material is nickel gold-plated kovar.
Cap is kovar.
Base is ceramic.
Package weight is 6.5 grams.

68 Lead Ceramic Leaded Chip Carrier



NOTES:

1. Lead Material Is Alloy 42 or Kovar Solder Coated
2. Lid Is Kovar or Alloy 42
3. Base & Top Are Ceramic
4. Cavity Size Is .400" × .400"
5. Package Weight Is 7.5 Grams
6. Lead Forming Is Optional

FAIRCHILD
A Schlumberger Company

Section 3
F8 Microcomputer Family

General

The distribution of logic among the various elements of a microcomputer system is one of the most variable features of such systems. The traditional division of logic corresponds to the requirements of a computer; e.g., one device serving as CPU, one as memory, and one as I/O. In the F8 microcomputer family, logic is implemented in devices in terms of application complexity rather than in terms of computer function. Thus, for example, two F8 devices implement all of the basic functions of a small microcomputer.

To accomplish this, the design of the F8 family includes a number of non-traditional function assignment features:

1. A small amount of RAM is implemented within the CPU as a scratchpad memory.
2. Memory addressing logic is implemented in the memory devices rather than in the CPU.
3. The I/O ports are implemented in the CPU and memory devices rather than in discrete I/O devices.

Every F8 configuration must contain an F3850 CPU, at least one F3851 Program Storage Unit (PSU) or memory interface device, and standard ROM or PROM (see figure 3-1). The memory-oriented devices may be used singly or together in the same system; when necessary, multiple units of the same type may be used. For example, an F3850 and two F3851s may comprise a system requiring 2K words of ROM, 64 bytes of RAM, and six I/O ports.

Memory Interface Devices

When required by the application, the F3851 PSU may be replaced by an F3853 Static Memory Interface (SMI). Both of these devices interpret control signals output by the F3850 and generate the standard address and control signals required by off-the-shelf dynamic and static memory devices.

Input/Output Devices

Applications that require additional I/O and interrupt capabilities but do not require the PSU storage capacity can make use of the F3861 Peripheral Input/Output (PIO) device. The PIO, which also contains interrupt logic and a programmable timer, interprets CPU control signals to drive two 8-bit I/O ports.

Bus Structure

The F8 microcomputer components are interconnected by means of a system bus structure that is composed of the following elements:

1. Eight data bus lines (DB_0 – DB_7)
2. Five control lines ($ROMC_0$ – $ROMC_4$)
3. Two clock lines (Φ, WRITE)
4. Three interrupt lines (PRI IN, PRI OUT, INT REQ)

Instruction Set

The instruction set of a microprocessor or microcomputer is the software tool used to shape the device or system for a particular application. The F8 instruction set is divided into four functional groups.

1. Input/Output
2. Arithmetic/Logical
3. Address Register Control
4. Indirect Scratchpad Address Register (ISAR) and Status Control

The F8 instruction set is presented in table 3-1.

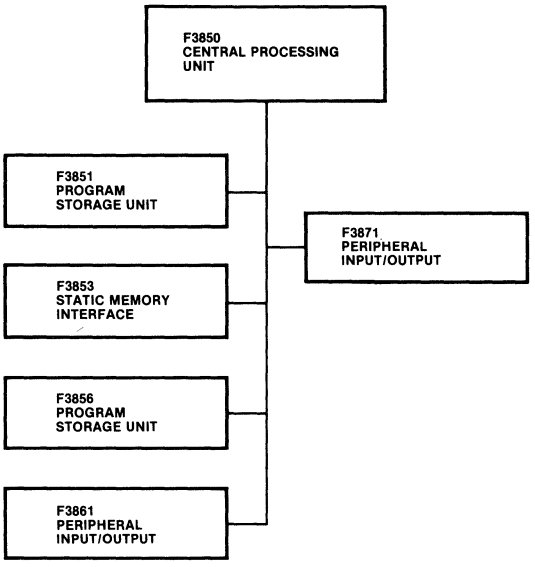
Table 3-1 F8 Instruction Set

Instruction	Description
ADC	Add Accumulator to Data Counter
AI	Add Immediate to Accumulator
AM	Add (Binary) Memory to Accumulator
AMD	Add (Decimal) Memory to Accumulator
AS	Add (Binary) Scratchpad Memory to Accumulator
ASD	Add (Decimal) Scratchpad Memory to Accumulator
BC	Branch on Carry
BF	Branch on false
BM	Branch on Negative
BNC	Branch if No Carry
BNO	Branch if No Overflow
BNZ	Branch if Not Zero
BP	Branch if Positive
BR	Unconditional Branch
BR7	Branch on ISAR
BT	Branch on True
BZ	Branch on Zero
CI	Compare Immediate
CLR	Clear Accumulator
COM	Complement
DCI	Load Data Counter Immediate
DI	Disable Interrupt
DS	Dessement Scratchpad Memory Content
EI	Enable Interrupt
IN	Input Long Address
INC	Increment Accumulator
INS	Input Short Address
JMP	Branch Immediate
LI	Load Immediate
LIS	Load Immediate Short
LISL	Load Lower Octal Digit of ISAR
LISU	Load Upper Octal Digit of ISAR
LM	Load Accumulator from Memory
LNK	Link Carry to Accumulator
LR	Load Register
NI	AND Immediate
NM	Logical AND from Memory
NOP	No Operation
NS	Logical AND from Scratchpad Memory
OI	OR Immediate
OM	Logical OR from Memory
OUT	Output Long Address
OUTS	Output Short Address
PI	Call to Subroutine Immediate
PK	Call to Subroutine Direct and Return from Subroutine Direct
POP	Return from Subroutine
SL	Shift Left
SR	Shift Right
ST	Store to Memory
XDC	Exchange Data Counters
XI	Ekxclusive-OR Immediate
XM	Exclusive-OR from Memory
XS	Exclusive-OR from Scratchpad Memory

Descriptions

Following is data that describes the members of the F8 microcomputer system family.

F8 FAMILY ORGANIZATION



FAIRCHILD
A Schlumberger Company

F3850

Central Processing Unit (CPU)

Microprocessor Product

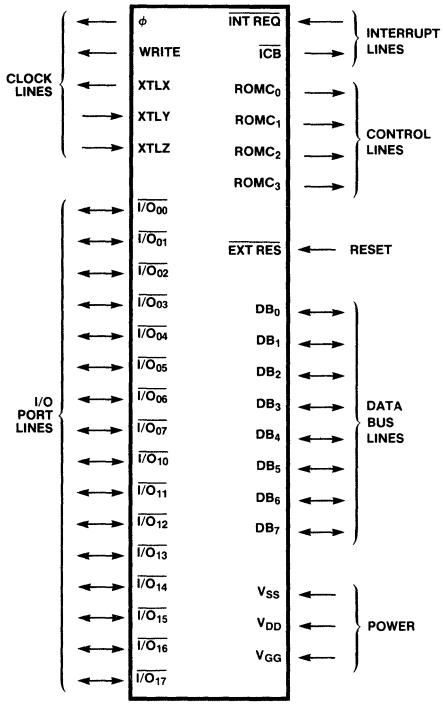
Description

The Fairchild F3850 is the Central Processing Unit (CPU) for the F8 8-Bit Microprocessor family. The F3850 contains more than 70 instructions in its instruction set and operates on 8-bit units of information.

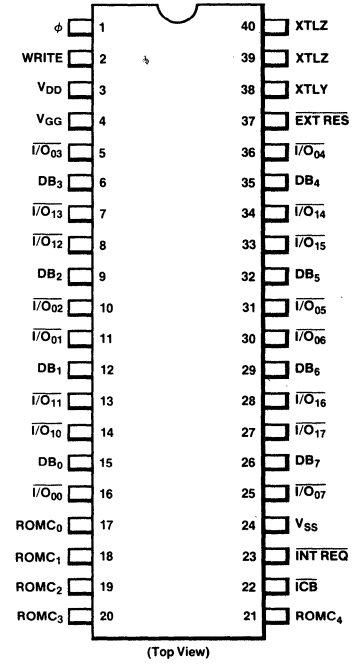
- **N-channel Isoplanar MOS Technology**
- **2 μs Cycle Time**
- **64-Byte Scratchpad on the CPU Chip**
- **Two Bidirectional, 8-Bit I/O Ports, with Output Latches**
- **8-Bit Arithmetic and Logic Unit, Supporting Both Binary and Decimal Arithmetic**
- **Interrupt Control Logic**
- **Power-on Reset Logic**
- **Clock Generation Logic Within the CPU Chip, With Crystal and External Clock Generation**
- **More Than 70 Instructions**
- **+5 V and +12 V Power Supplies**
- **Low Power Dissipation (Typically Less Than 330 mW)**

3

Signal Functions



Connection Diagram



(Top View)

Device Organization

The logical organization and pins for the F3850 CPU are illustrated in *Figure 1.*

Arithmetic and Logic Unit

The arithmetic and logic unit (ALU) provides all data manipulating logic for the F3850. It contains logic that operates on a single 8-bit source data word or combines two 8-bit words of source data to generate a single 8-bit result. Additional information is reported in status flags, where appropriate.

Operations performed on two units of source data include addition, compare, and the Boolean operations (AND, OR, Exclusive-OR). The two sources are input to the ALU through the left and right multiplexer buses; the result is placed on the result bus.

Operations performed on a single 8-bit unit of source data include complement, increment, decrement, shift right, shift left, and clear. The source is input to the ALU through either the left or right multiplexer bus; the result is placed on the result bus.

Instruction Register

The CPU contains registers for storing various types of data. The instruction register holds an 8-bit code, which defines the operations to be performed by the CPU.

The contents of the instruction register are decoded by control unit logic, which generates signals to enable specific sequences of logic operations within the CPU chip. In response to the contents of the instruction register, the control unit also generates five signals, $ROMC_0$ through $ROMC_4$, that control operations throughout the microprocessor system.

Accumulator

The accumulator is a general-purpose 8-bit data register, which is the most common data source and results destination for the ALU.

Scratchpad and ISAR

The scratchpad provides 64 8-bit registers that may be used as general-purpose RAM memory (see *Figure 2*).

Figure 2 F8 Programming Model

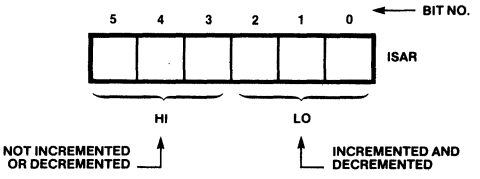
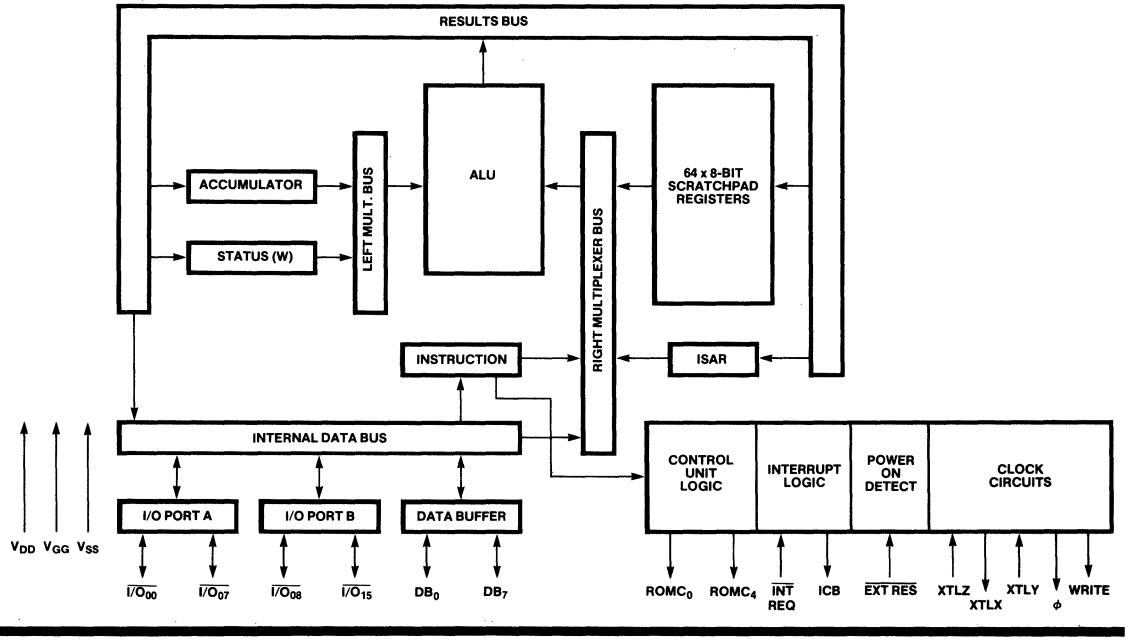


Figure 1 F3850 CPU Logical Organization

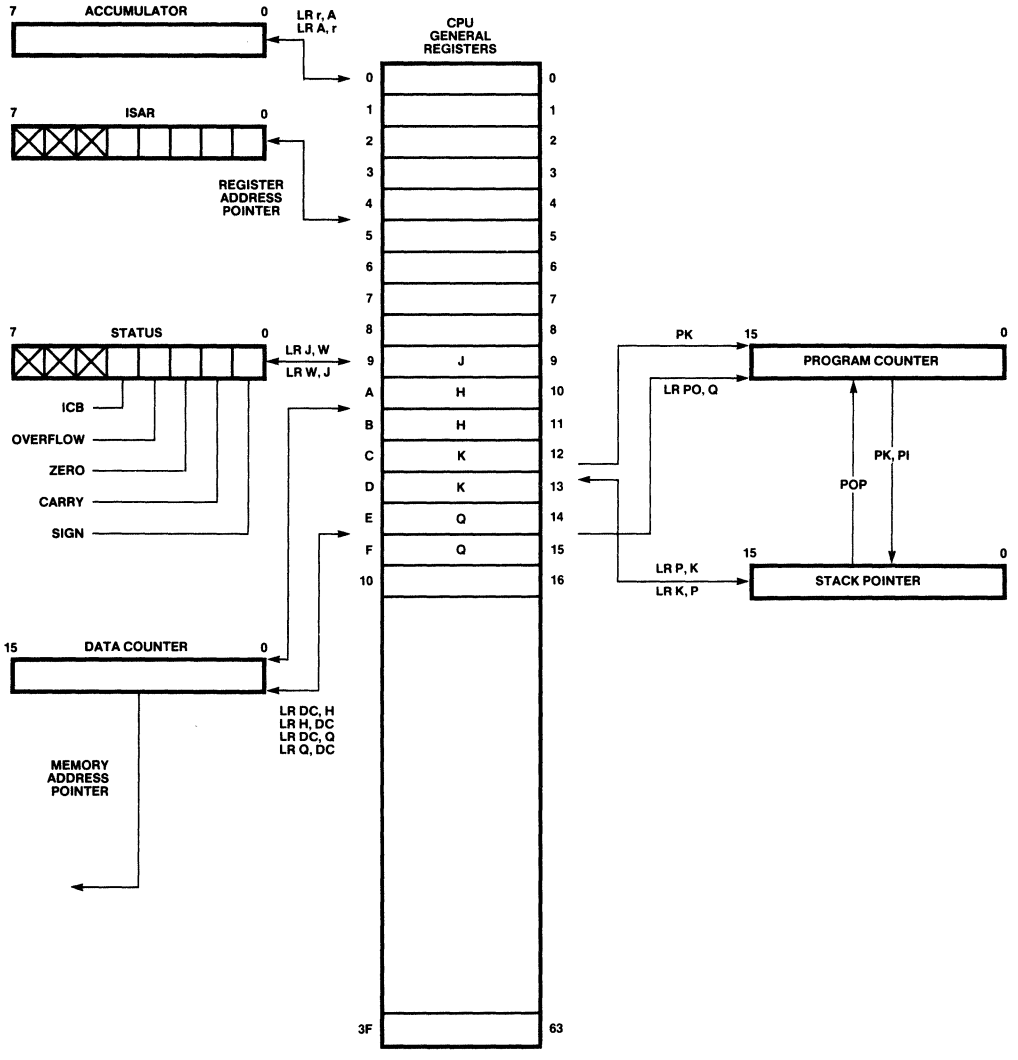


The indirect scratchpad address register (ISAR) is a 6-bit register used to address the 64 scratchpad registers.

The first 16 scratchpad bytes can be identified either by instructions without using the ISAR or referenced through the ISAR. The remaining scratchpad bytes are referenced through the ISAR; i.e., the ISAR is assumed to hold the address of the scratchpad byte that is to be referenced.

The ISAR may be visualized as holding two octal digits, HI and LO, as illustrated in *Figure 3.* This division of the ISAR is important, since a number of instructions increment or decrement the con-

Figure 3 ISAR Register



tents of the ISAR, when referencing scratchpad bytes through the ISAR. This makes it easy to reference a buffer consisting of contiguous scratchpad bytes. However, only the low-order octal digit (LO) is incremented or decremented; thus ISAR is incremented from O'27'* to O'20', not to O'30'. Similarly, ISAR is decremented from O'20' to O'27', not to O'17'. This feature of the ISAR is very useful in that it greatly simplifies many program sequences.

Selected scratchpad registers are reserved for direct communication with other registers within the F8 system, as illustrated in *Figure 4.*

Scratchpad register 9 (O'11') is used as temporary storage for the CPU status register (W register). Scratchpad registers 10 through 15 (O'12' through O'17') communicate directly with data and program memory address registers that are maintained on the F3851, F3852, and F3853 chips. *Figure 4* identifies the data transfers that can be implemented by executing a single F8 instruction. For example, the illustration:

W register of F3850 CPU ↔ J

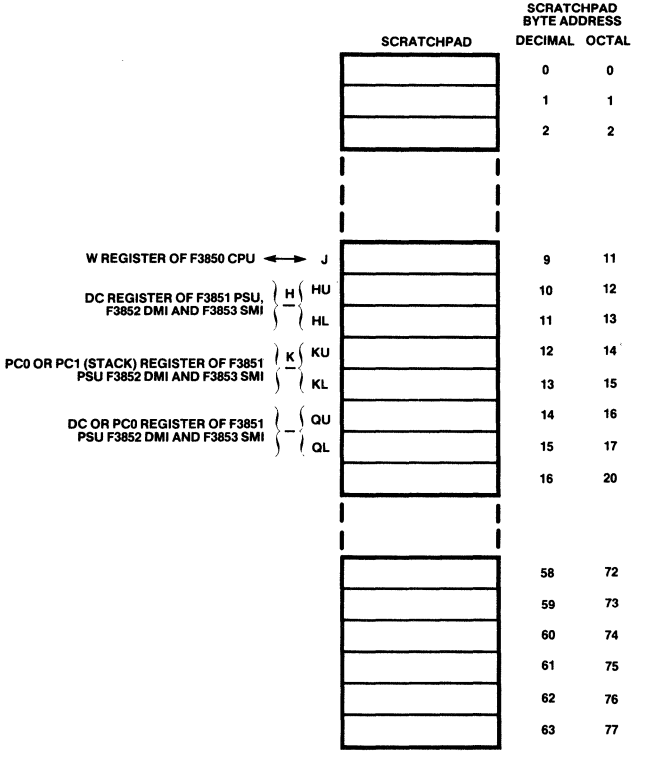
means that a single instruction can move the contents of the W (or status) register to scratchpad register 9 (J register). Another single instruction can move data in the opposite direction.

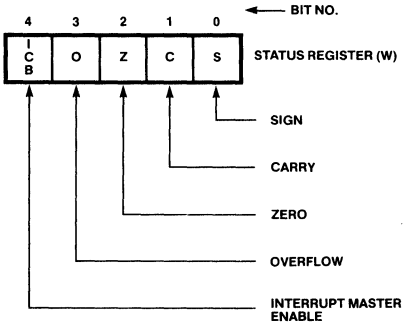
Status Registers

The status (W) register holds five status flags. *Table 1* summarizes the way each flag is used. Note that status flags are selectively modified following execution of different instructions. See the "Instruction Execution" section for a discussion of the way individual F8 instructions modify status flags.

*The notation O'nn' represents an octal number.

Figure 4 F3850 CPU Scratchpad Registers





Sign (S Bit)—When the results of an ALU operation are being interpreted as a signed binary number, the high-order bit (bit 7) represents the sign of the number. At the conclusion of instructions that may modify the accumulator bit 7, the S bit is set to the complement of the accumulator bit 7.

Table 1 Summary of Status Bits

OVERFLOW = $CARRY_7 + CARRY_6$
ZERO = $\overline{ALU_7}\ \overline{ALU_6}\ \overline{ALU_5}\ \overline{ALU_4}\ \overline{ALU_3}\ \overline{ALU_2}\ \overline{ALU_1}\ \overline{ALU_0}$
CARRY = $CARRY_7$
SIGN = $\overline{ALU_7}$

Carry (C Bit)—The C bit may be visualized as an extension of an 8-bit data unit; i.e., the ninth of a 9-bit data unit. When two bytes are added, and the sum is greater than 255, then the carry out of the high-order bit appears in the C bit; e.g.:

	C 7 6 5 4 3 2 1 0 ← Bit Number
Accumulator contents:	0 1 1 0 0 1 0 1
Value added:	0 1 1 1 0 1 1 0
Sum:	0 1 1 0 1 1 0 1 1

There is no carry, so C is reset to 0.

	C 7 6 5 4 3 2 1 0 ← Bit Number
Accumulator contents:	1 0 0 1 1 1 0 1
Value added:	1 1 0 1 0 0 0 1
Sum:	1 0 1 1 0 1 1 1 0

There is a carry, so C is set to 1.

Zero (Z bit)—The Z bit is set whenever an arithmetic or logical operation generates a zero result. The Z bit is reset to 0 when an arithmetic or logical operation could have generated a zero result but did not.

Overflow (O Bit)—When the results of an ALU operation are being interpreted as a signed binary number, since the high-order bit (bit 7) represents the sign of the number, some method must be provided for indicating a carry out of the highest numeric bit (bit 6). This is done using the O bit. After arithmetic operations, the O bit is set to the Exclusive-OR of a carry out of bits 6 and 7. The simplification of signed binary arithmetic is described in the *F8 and F3870 Guide to Programming*; examples are presented below:

	7 6 5 4 3 2 1 0 ← Bit Number
Accumulator contents:	1 0 1 1 0 0 1 1
Value added:	0 1 1 1 0 0 0 1
Sum:	1 1 1 0 0 1 0 0

There is a carry out of bit 6 and a carry out of bit 7, so the O bit is reset to 0 ($1 \oplus 1 = 0$). The C bit is set to 1.

	7 6 5 4 3 2 1 0 ← Bit Number
Accumulator contents:	0 1 1 0 0 1 1 1
Value added:	0 0 1 0 0 1 0 0
Sum:	1 0 0 0 1 0 1 1

There is a carry out of bit 6, but no carry out of bit 7; the O bit is set to 1 ($1 \oplus 0 = 1$). The C bit is reset to 0.

Interrupts (ICB Bit)—External logic can alter program execution sequence within the CPU by interrupting ongoing operations. However, interrupts are allowed only when the ICB is set to 1; interrupts are disallowed when the ICB is reset to 0.

Control Unit
The control unit decodes the contents of the instruction register and generates two sets of control signals. These signals are transparent to the user.

Five control signals ($ROMC_0$ through $ROMC_4$) are output by the control unit to identify operations that other chips of the F8 family must perform. These signals are described in the "ROMC Signals" section.

Interrupt Logic
This logic handles the interrupt requests. For a complete description refer to the "Interrupt" discussion within the "Instruction Execution" section.

Power on Detect
When the External Reset ($\overline{EXT\ RES}$) signal is pulled low and then returned high, or when power is turned on, the power on detect logic sets the PC registers to 0, causing a program originating at memory location 0 to be executed. Also, the interrupt control status bit is set low, inhibiting interrupt acknowledgement. The system is locked in an idle state while $\overline{EXT\ RES}$ is held low.

Signal Descriptions

The F3850 input and output signals are described in *Table 2.*

Table 2 F3850 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
ϕ	1	Clock	These output signals drive all other devices in the F8 family.
WRITE	2	Write	
XTLX	39	Crystal Clock	The XTLX output signal is used when generating the system clock in the crystal mode (with the XTLY and XTLZ signals).
XTLY	38	External Clock	The XTLY input signal is used with the XTLX signal when generating the system clock in the crystal mode, and is also used for operating in the external clock mode.
XTLZ	40	Crystal Clock	This input signal must be grounded for crystal clock or external clock.
I/O Port			
$\overline{I/O}_{00}$-$\overline{I/O}_{07}$	16, 11, 10, 5, 36, 31, 30, 25	I/O Port Zero	These bidirectional signals are ports through which the CPU communicates with logic external to the microprocessor system.
$\overline{I/O}_{10}$-$\overline{I/O}_{17}$	14, 13, 8, 7, 34, 33, 28, 27	I/O Port One	
Interrupt			
$\overline{ICB}$	22	Interrupt Control Bit	The $\overline{ICB}$ output signal indicates whether or not the CPU is currently ignoring the $\overline{INT\ REQ}$ line. If the $\overline{ICB}$ signal is low, the CPU responds to interrupt requests; if the $\overline{ICB}$ signal is high, the CPU ignores interrupt requests.
$\overline{INT\ REQ}$	23	Interrupt Request	This input line is used to signal the CPU that an interrupt is being requested.The F3851 PSU, F3861 and F3871 PIOs, and F3853 SMI devices contain logic to initiate interrupt requests by pulling the $\overline{INT\ REQ}$ signal low. The CPU acknowledges interrupt requests by outputting the appropriate ROMC signals.
Control			
$ROMC_0$-$ROMC_4$	17—21	Control	The ROMC output signals control logic operations for other devices in the F8 family. These signals assume a state early in each machine cycle and hold that state for the duration of the cycle. Refer to the "Instruction Execution" section for further discussion and a summary table of the ROMC interpretation by CPU logic.

Table 2 F3850 Signal Descriptions (Continued)

Mnemonic	Pin No.	Name	Description
Reset			
$\overline{EXT}$ $\overline{RES}$	37	External Reset	This input signal can be used to externally reset the system. When the line is pulled low, a program originating at memory address 0 is executed.
Data Bus			
DB_0–DB_7	15, 12, 9, 6, 35, 32, 29, 26	Data Bus	These eight bidirectional signals are data bus lines that link the F3850 CPU with all other F8 devices in the system. They are multiplexed lines used to transfer data and addresses.
Power			
V_{DD}	3	Power Supply	Nominal +5 Vdc
V_{GG}	4	Power Supply	Nominal +12 Vdc
V_{SS}	24	Ground	Common power and signal return

3

Clock Circuits

A unique feature of the F8 microprocessor is that clock logic forms an integral part of the F3850 CPU chip. The F3850 CPU offers two methods of generating a system clock: crystal mode and external mode.

Crystal Mode

Figure 5 shows the pin configuration for clock generation using the crystal mode. A crystal in the 1- to 2-MHz range is placed across the XTLX and XTLY pins, along with two capacitors (C_1 and C_2), to provide a highly precise clock frequency. The external crystal (and capacitors) together with internal circuitry combine to form a parallel resonant crystal oscillator. Capacitors C_1 and C_2 should be approximately 15 pF. The characteristics of the crystal used in this mode of clock generation are summarized as:

Frequency: 1 to 2 MHz, typical AT cut
Mode of Oscillation: Fundamental
Operating Temp. Range: 0°C to +70°C
Drive Level: 10 mW
Frequency Tolerance: $f_0 = 1$ or 2 MHz $\pm$1000 ppm @ C_L = 20 pF

External Mode

For F8 applications where synchronization with an external system clock is desired, the external clock mode may be used as shown in *Figure 6*. For example, a slave F3850 CPU may receive its timing from a master F3850 CPU by having the master ϕ output drive the slave XTLY input.

Figure 5 Crystal Mode Clock Generation

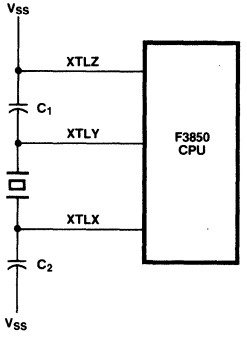


Figure 6 External Mode Clock Generation

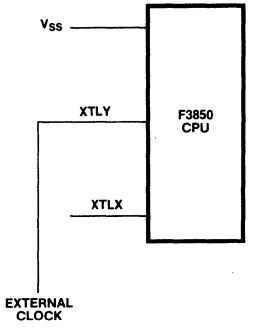


Figure 7 illustrates the timing characteristics of the clock signal needed for external mode clock generation and the timing characteristics of the ϕ and WRITE signals generated by the CPU.

Timing Signal Outputs
In response to the three clock mode inputs, the F3850 CPU outputs two timing signals: clock signal ϕ and instruction cycle control signal WRITE. As shown in *Figure 7*, ϕ is the signal used to synchronize the entire microprocessor system. The WRITE signal defines the duration of each machine cycle. Refer to the "Instruction Execution" section. Parameters and specifications for the timing signals are detailed in the "Timing Characteristics" section.

Instruction Execution

The F3850 CPU logic controls instruction execution through the ϕ and WRITE timing signals, plus the five ROMC control lines. Devices external to the F3850 CPU must respond directly to these signals.

Instruction Cycle
All instructions are executed in cycles that are timed by the trailing edge of WRITE.

There are two types of instruction cycle: the short cycle, which is four ϕ periods long, and the long cycle, which is six ϕ periods long. The long cycle is sometimes referred to as 1.5 cycles. *Figure 7* illustrates the short cycle (PW_S) and the long cycle (PW_L). Note that WRITE high appears only at the end of an instruction cycle.

The simplest instructions of the F8 instruction set execute in one short cycle. The most complex instruction (PI) requires two short cycles plus three long cycles.

ROMC Signals
The CPU logic uses the five ROMC signals to identify operations that devices must perform during any instruction cycle. The 32 possible ROMC states are described in the "ROMC Signal Functions" section. The state of the ROMC signals and the operation they identify last through one instruction cycle.

The general distribution of logic among devices of the F8 family and general data movements associated with instruction execution are given in the *F8 and F3870 Guide to Programming*.

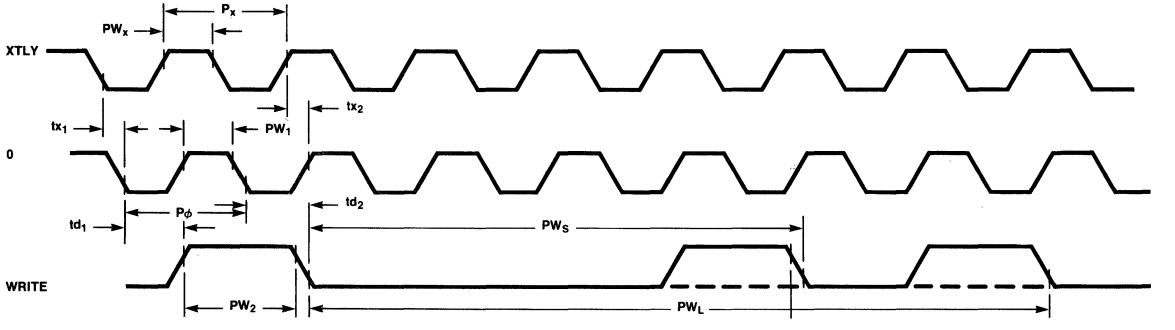
Memory addressing logic is located on the F3851 Program Storage Unit (PSU), the F3852 Dynamic Memory Interface (DMI), and the F3853 Static Memory Interface (SMI) devices. Each of these devices contains registers to address programs (PC0 and PC1) or data (DC0 or DC1). The F3851 PSU does not have a DC1 register.

Unlike other microprocessors, the F3850 CPU does not output addresses at the start of memory access sequences; a simple command to access the memory location addressed by PC0 or DC0 is sufficient, since the device receiving the memory access command contains PC0 and DC0 registers. (The PC1 and DC1 are buffer registers for PC0 and DC0.)

Moving memory addressing logic from the CPU to memory (and memory interface) devices simplifies CPU logic; however, it creates the potential for devices to compete when responding to memory access commands.

There will be as many PC0 and DC0 registers in a microcomputer system as there are PSU, DMI, and SMI devices; the ambiguity of which unit will respond to a memory read or write command is resolved by ensuring that all PC0 and DC0 registers contain the same information at all times. Every PSU, DMI, and SMI device

Figure 7 Clock Generation Timing Signals



has a unique address space, i.e., a unique block of memory addresses within which it responds to memory access commands.

For example, an F3851 PSU may have an address space of H'0000' through H'03FF'; an F3852 DMI may have an address space of H'0400' through H'07FF'. If a microcomputer system has these two memory devices and no others, then the F3851 PSU will respond to memory access commands when the PC0 or DC0 registers (whichever are identified as the address source) contain a value between H'0000' and H'03FF'; the F3852 DMI will respond to addresses in the range H'0400' through H'07FF'. No device will respond to addresses beyond H'07FF', even though such addresses may exist in PC0 and/or DC0.

Each device compares its address space with the contents of PC0 and DC0, whichever is identified as the address source, and only responds to a memory access command if the contents of PC0 or DC0 is within the device's address space.

If all memory address registers (PC0, PC1, DC0, and DC1) are to contain the same information, then ROMC states that require any of these registers' contents to be modified must be acted upon by all devices containing any of these four registers. If devices are not to compete when an ROMC state specifies that a memory access must be performed, then only a device whose address space includes the identified memory address must respond to the ROMC state.

As illustrated in *Figure 8,* the five ROMC signals that define the ROMC state are output early in the instruction cycle and are maintained stable for the duration of the instruction cycle; i.e., only one ROMC state can be specified per instruction cycle. Therefore, devices can only be called upon to perform one instruction execution related operation per one instruction cycle.

As referenced in the "ROMC Signal Functions" section, each ROMC state is identified by individual signal line states (1 for high, 0 for low), and by a two-digit hexadecimal code. The hexadecimal code is used to identify ROMC states throughout this data sheet. Also given in the "ROMC Signal Functions" section is the instruction cycle length (short or long) implied by each code, plus the way in which codes must be interpreted by the other F8 devices.

Instruction Execution Sequence

Every instruction execution sequence ends with an instruction code being fetched from memory to identify the next instruction cycle. The instruction code is loaded into the CPU instruction register, out of which it is decoded by the CPU control unit logic. An instruction fetch is executed during the last instruction cycle of the previous instruction, as illustrated in *Figure 9.*

There is a group of F8 instructions that cause operations to occur entirely within the F3850 CPU. These instructions do not use the data bus, therefore can execute in one cycle. Since one-cycle instructions do not use the data bus, no ROMC state needs to be generated for the one-cycle instruction being executed; therefore, as illustrated in *Figure 9,* ROMC state 0 is specified, causing the instruction fetch of the next instruction.

Multi-cycle instructions must end with a cycle that does not use the data bus; ROMC state 0 is specified at the beginning of this last instruction cycle, causing the next instruction to be fetched.

Following an instruction fetch, CPU logic decodes the fetched instruction code and executes the specified instruction. There are Five types of instruction cycles that can follow.

1. Operations may all be internal to the CPU. This will be the last or the only cycle for an instruction, and will specify ROMC state 0, as illustrated in *Figure 9.*

Figure 8 ROMC Timing Signals Output by F3850 CPU

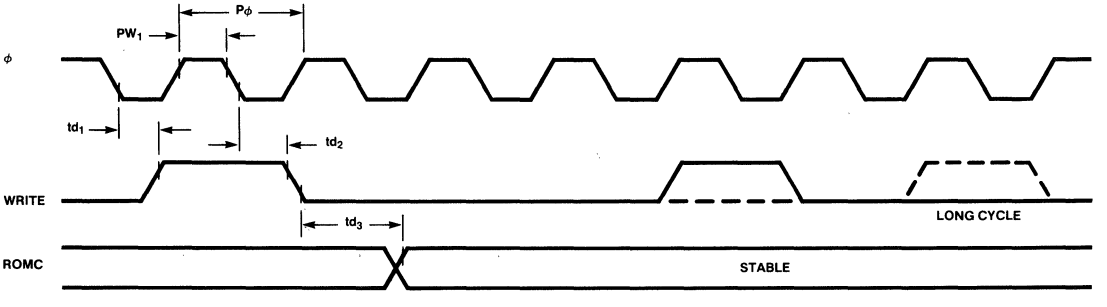


Figure 9A Short Cycle Instruction Fetch

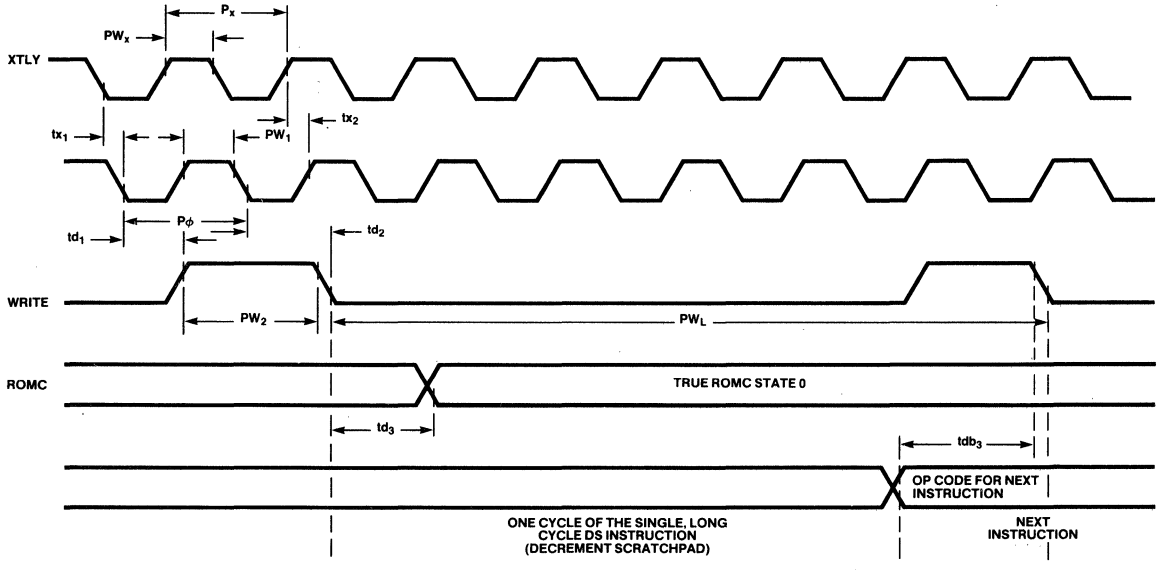
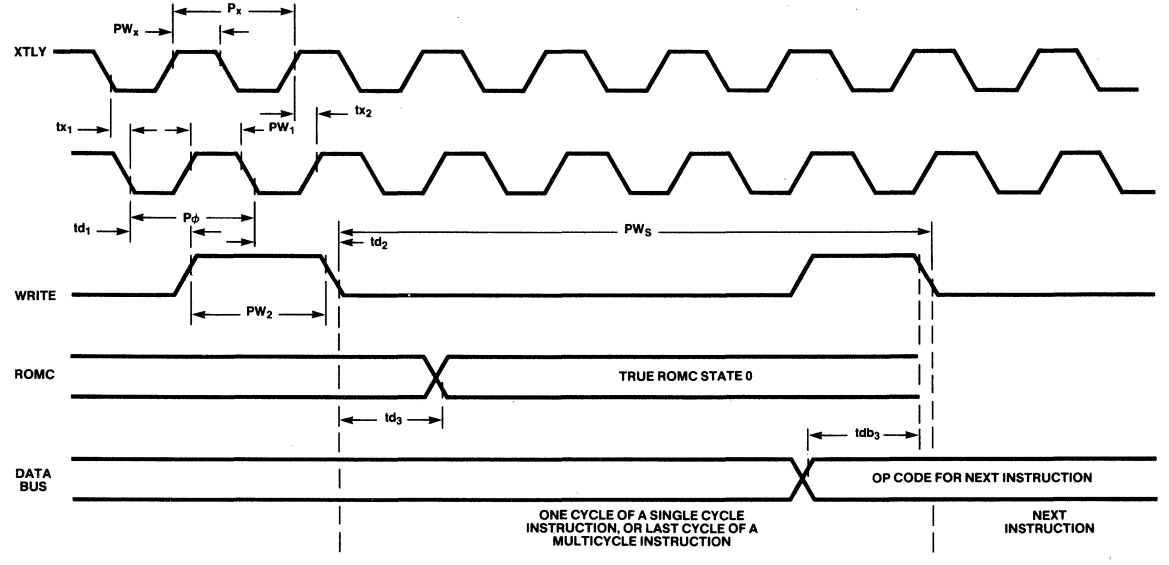


Figure 9B Long Cycle Instruction Fetch (During DS Only)



2. Data may be transferred between the F3850 CPU and memory devices. See the "Referencing Memory" section.
3. Data may be transferred from one memory device to all memory devices. The CPU is not the transmitter or the receiver of data in this transfer. See the "Memory-to-Memory Data Transfers" section.
4. Data may be transferred to or from an I/O port, as described in the "Input/Output Interfacing" section.
5. An interrupt may be acknowledged, as described in the "Interrupts" section.

Every F8 instruction is executed as one, or a sequence of, standard instruction cycles. Timing for the standard instruction cycles is illustrated in *Figures 9, 10, 11* and *12*.

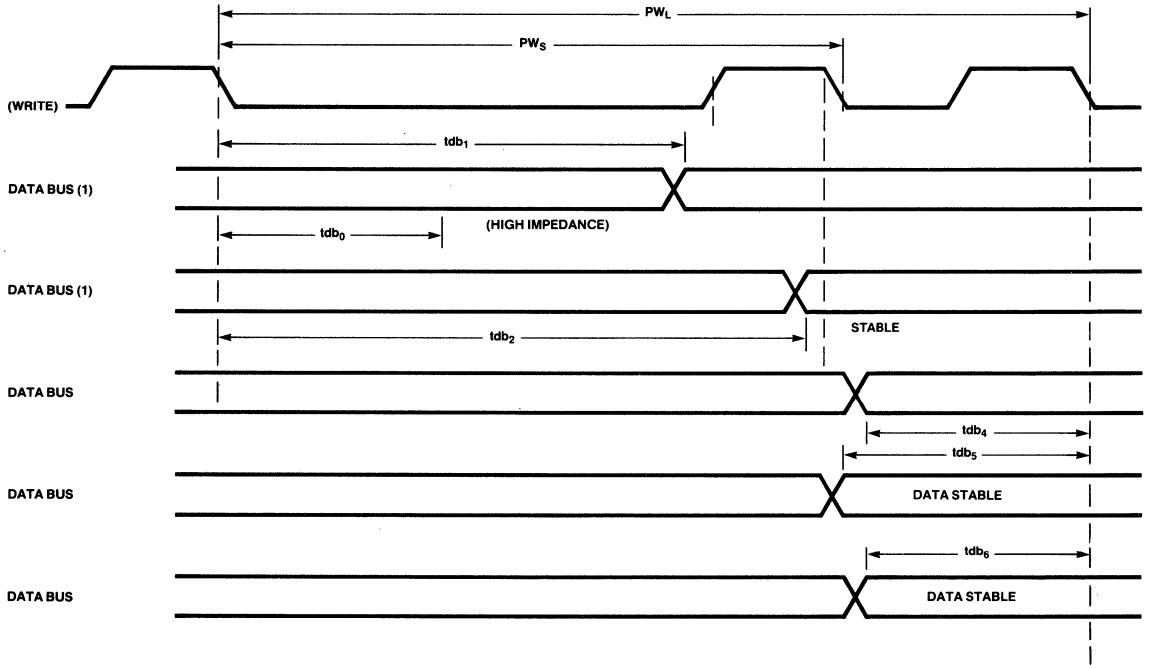
Refer to the "Instruction Cycle Execution and Timing" section for a list of the instruction cycles and their associated ROMC state.

Referencing Memory

Memory may be referenced during an instruction cycle either to transfer the data from the CPU to a memory word or to transfer data from a memory word to the CPU. A memory reference occurs as shown in *Figure 10*.

If data is being output by the CPU, then the delay before data output is stable will be tdb_1 when data comes from the accumulator; the instruction cycle will be long. The delay before data output is stable will be tdb_2 when data comes from the scratchpad; the instruction cycle in this case will also be long.

Figure 10 Memory Reference Timing



(1) Timing for CPU outputting data onto the data bus.

Delay tdb_1 is the delay when data is coming from the accumulator.

Delay tdb_2 is the delay when data is coming from the scratchpad (or from a memory device).

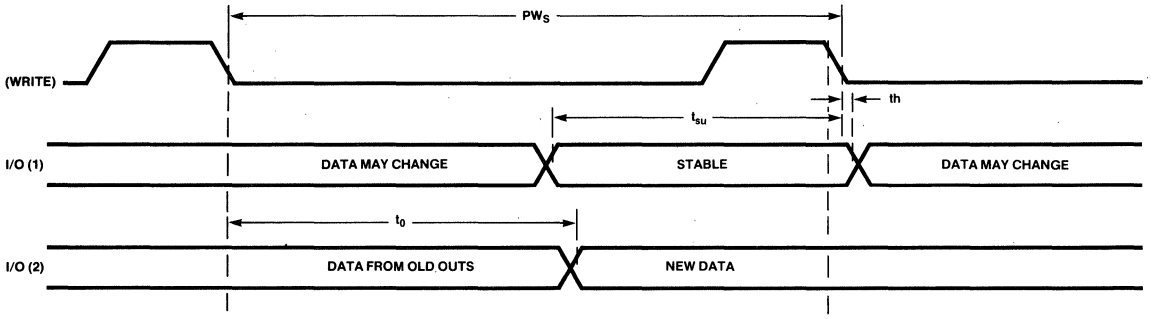
Delay tdb_0 is the delay for the CPU to stop driving the data bus.

(2) There are four possible cases when inputting data to the CPU, via the data bus lines which depend on the data path and the destination in the CPU, as follows:

tdb_3: Destination — IR (instruction Fetch)
tdb_4: Destination — Accumulator (with ALU operation — AM)
tdb_5: Destination — Scratchpad (LR K,P etc.)
tdb_6: Destination — Accumulator (no ALU operation — LM)

In each case a stable data hold time of 50 ns from the WRITE reference point is required.

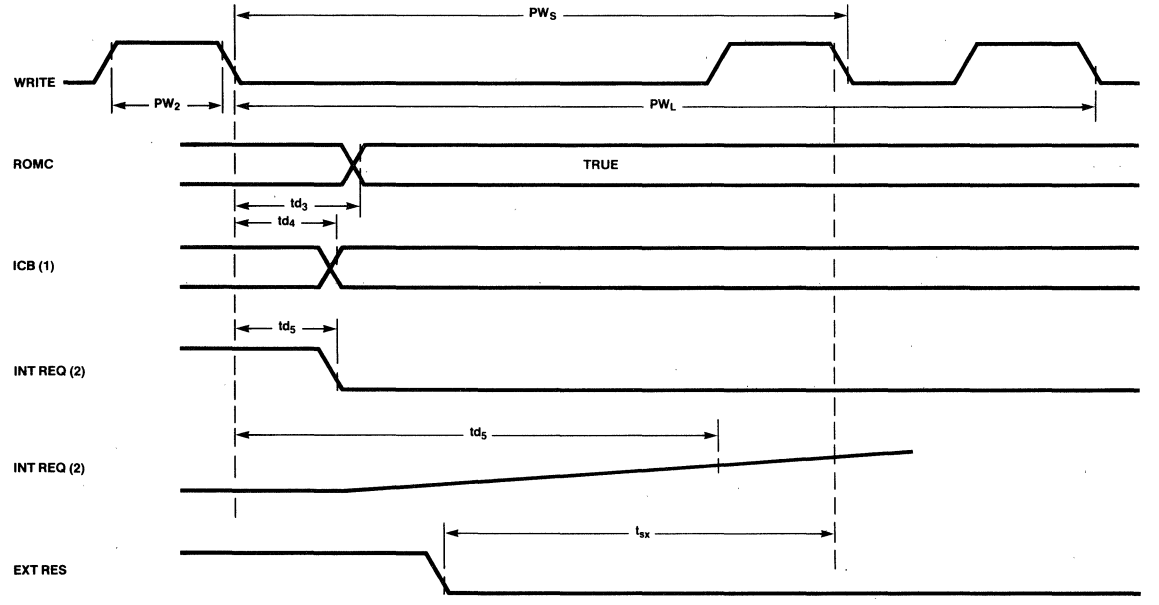
Figure 11 Timing for Data Input or Output at I/O Port Pins



(1) This represents the timing for data at the I/O pin during the execution of the INS instruction, i.e., the CPU is inputting.

(2) This represents the timing for data being output by the CPU at the I/O pin.

Figure 12 Interrupt Signals Timing



(1) The $\overline{ICB}$ signal will go from a 1 to a 0 following the execution of the E1 instruction and will go from a 0 to a 1 following either the execution of the D1 instruction or the CPU's acknowledgement of an interrupt.

(2) This is an input to the CPU chip and is generated by a PSU or F3853 M1 chip. The open drain outputs of these chips are all wire—ANDed together on this line with the pull-up being located on the CPU chip. For a 0 to 1 transition the delay is measured to 2.0 V.

If data is being input to the CPU, then the delay before incoming data must be stable depends on the destination of the data, as illustrated in *Figure 10.*

The type of data transfer is identified by the ROMC state that is output at the beginning of the instruction cycle.

The instruction fetch may also be viewed as a memory reference operation where the destination is the instruction register. Timing for this case is illustrated in *Figure 9.*

Memory-to-Memory Data Transfers
In response to appropriate ROMC states, data can be transferred from one memory device to all memory devices during one instruction cycle. For example, data can be transferred from a memory byte within (or controlled by) one memory device, to one byte of an address register (PC0 or DC0) within all memory devices.

Three ROMC states (C, E, and 11) specify operations of this type, and *Figure 10* illustrates timing for the data transfer. In *Figure 10,* tdb_2 is the delay until data from memory or a memory address register is stable on the data bus.

Input/Output Interfacing
Programmed I/O in the F8 microcomputer system is influenced by the design of the I/O port pins. As illustrated in *Figure 13*, each I/O port pin is a "wire-AND" structure between an internal latch and an external signal, if any. The latch is always loaded directly from the accumulator.

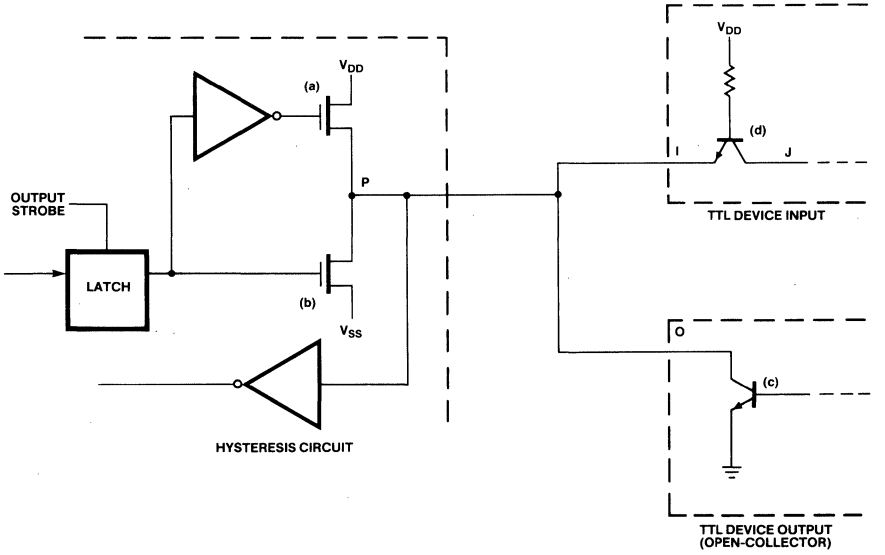
Each F8 I/O pin can be set high or low under program control. If a 1 (high) is presented at the latch, then gate (b) turns on and gate (a) turns off, so that P is at V_{SS} (low). If a 0 (low) is presented at the latch, then gate (a) turns on and gate (b) turns off, so that P is at V_{DD} (high).

When outputting data through an I/O port, the pin can be connected directly to a TTL gate input ("TTL Device Input" in *Figure 13*). Data is input to the pin from a "TTL Device Output" in *Figure 13.*

In normal operation, high or low levels at P drive the external TTL device input transistor (d). If a low level is set at P, transistor (d) conducts current through the path J, 1, P, and FET(b). This is transferred as a low level to the rest of the circuits in the TTL device and results in a high or low level at the output of the device, depending on its characteristics. If the level at P is set high, transistor (d) does not conduct current, and a high level is transferred by (d).

When data is input to the I/O pin, high or low levels at 0 drive the hysteresis circuit in the port and result in logic ones or zeros being transferred to the accumulator.

Figure 13 F8 I/O Port Bit



Since the I/O pin and the TTL device output at 0 are wire-ANDed, it is possible for the state of one to affect the transfer of data out from the I/O pin or in from the TTL device output. For example, if the latch in the I/O port is set so that the pin is clamped low by (b), then the level at 0 cannot pull P high. Conversely, if P is clamped to a low level by (c), setting the latch for a high level has no effect.

All I/O port bits should be set for a high level, before data input, to prevent incoming logic zeros from being "masked" by logic ones present at the port from previous outputs. In some instances, the ability to mask bits of a port to logic 1 is useful. (Note that logic 1 becomes a 0 V electrical level at the I/O pin; logic 0 corresponds to a high electrical level.)

The F8 CPU can execute two types of programmed I/O operation:

1) I/O via the two CPU ports (0 and 1)
2) I/O via ports on the other devices

Input/Output operations that use the two CPU I/O ports execute in two instruction cycles. During the first cycle, the fetched instruction is decoded; the data bus is unused. In this cycle data is either sent from the accumulator to the I/O latch or enabled from the I/O pin to the accumulator, depending on whether the instruction is an output or an input. At the falling edge of the WRITE signal (marking the end of the first cycle and beginning of the second cycle), the data is strobed into either the latch (OUTS) or the accumulator (INS), respectively. The second cycle is then used by the CPU for its next instruction fetch. *Figure 11* illustrates I/O timing.

Note that for the data input (INS) the setup and hold times specified are with respect to the WRITE pulse occurring at the end of the first cycle in the two-cycle instruction. For output data (OUTS) the delay is specified with respect to the falling edge of the WRITE signal marking the beginning of the second cycle in the two-cycle instruction.

Input/Output instructions that address I/O ports with an I/O port address greater than H'OF' occupy two bytes; the first byte specifies an IN or OUT instruction, while the second byte provides the I/O port address. Required timing at I/O port pins is given in the section of this data sheet that describes the device containing the addressed I/O port.

Interrupts

There are three CPU signals with interrupt processing; timing for all signals is illustrated in *Figure 12.*

An interrupt sequence is initiated by pulling either the $\overline{\text{INT REQ}}$ signal or the $\overline{\text{EXT RES}}$ signal low. In the case of the $\overline{\text{INT REQ}}$ signal nothing happens unless the $\overline{\text{ICB}}$ signal is low. Also, nothing happens until the next interruptable instruction comes to the end of execution. In the case of the $\overline{\text{EXT RES}}$ signal, execution of the interrupt routine begins in the machine cycle immediately following that in which the signal goes low, provided that the setup time specified in *Figure 12* has been met. The $\overline{\text{EXT RES}}$ signal response logic ignores the $\overline{\text{ICB}}$ signal.

In response to the $\overline{\text{INT REQ}}$ signal being low, when the CPU acknowledges the interrupt, it forces the $\overline{\text{ICB}}$ signal high and initiates instruction cycles with ROMC states 1C, 0F, 13, and 00, in that order. This causes program execution to branch to the interrupting device's address vector.

In response to the $\overline{\text{EXT RES}}$ signal being low, when the CPU acknowledges the interrupt, it forces the $\overline{\text{ICB}}$ signal high, then initiates instruction cycles with ROMC states 1C, 08, and 00, in that order. This causes program execution to branch to memory location 0.

The $\overline{\text{ICB}}$ signal is pulled low by the E1 instruction and is returned high by the D1 instruction.

Instruction Set Summary

The F3850 CPU instruction set is summarized in *Table 3.* This section does not attempt to give complete directions for programming the F8 microcomputer system; it explains signals and timing associated with the execution of every instruction. Refer to *F8 and F3870 Guide to Programming* for programming details. The columns used in *Table 3* are described below.

Op Code—The Op Code is the instruction mnemonic that appears in the mnemonic field of an assembly language instruction and identifies the instruction.

Operand (s)—If the instruction contains any information in the operand field of the assembly language source code, the information is shown in this column. Arrows identify the portion of object code that represents the operand field. Any portion of object code that does not represent the operand field must represent the mnemonic field. *Table 4* explains symbology used in the operand field.

Object Code—This is the hexadecimal representation of the instruction's object code. The first byte of object code, or in some cases the first hexadecimal digit of object code, represents the Op Code. The operand is represented by the second and third bytes of object code, if present, or in some cases by the second hexadecimal digit of the first object code byte. Refer to *Table 4* for symbology used in the object code field.

Cycle—This column identifies each instruction cycle for every instruction. Every cycle is listed on a separate horizontal line and is identified by the letter S for a short (four clock period) cycle or

the letter L for a long (six clock period) cycle. Thus, the entry

S

represents an instruction that executes in one short cycle. The entry

S
L
S

represents an instruction that executes in three cycles: the first is a short cycle; the second is a long cycle; the third (and last) is a short cycle.

ROMC State—This is the state, as identified in the "ROMC Signal Functions" section, that is output by the F3850 CPU in the early stages of the instruction cycle.

Timing—Timing for all instructions, except INS and OUTS accessing I/O ports 0 and 1, can be created out of *Figures 9* and *10*. For the exceptions, *Figure 11* is required.

The ROMC lines are always set after a delay of td_3, as shown in *Figure 9*. The only timing variations for each instruction cycle are data bus timing variations. Therefore, data bus timing is defined using the delays tdb_1 through tdb_6. With the exception of tdb_3, these time delays are unambiguous in that they are keyed to either the leading edge or the trailing edge of the WRITE signal high, for a long or short instruction cycle, as illustrated in *Figure 10*. There are two cases for tdb_3, however, as illustrated in *Figure 9*. These are identified in *Table 4* as 3S for *Figure 9A* and 3L for *Figure 9B;* tdb_1 through tdb_6 are otherwise identified by the numbers 1 through 6.

Cycles that do not use the data bus are identified by 0 in the timing column; *Figure 8* illustrates timing in this case.

Cycle	Represents
0	*Figure 8*
1	tdb_1 in *Figure 10*
2	tdb_2 in *Figure 10*
3S	tdb_3 in *Figure 9A*
3L	tdb_3 in *Figure 9B*
4	tdb_4 in *Figure 10*
5	tdb_5 in *Figure 10*
6	tdb_6 in *Figure 10*

Status Flags—Status flags are identified as follows:
O—Overflow
Z—Zero
C—Carry
S—Sign

Within each column, symbology is used as follows:

—	Status not affected
0	Status set to 0
I/O	Status set to either 1 or 0, depending on the results of the instruction's execution

Interrupt—An "x" in this column identifies an instruction that disallows interrupts at the end of the instruction's execution. A "y" identifies cycles in which the ICB is reset to 0 (cleared).

Function—The effect of each instruction cycle is described in this column using symbology given in *Table 4*.

Instruction Cycle Execution and Timing

Table 3 lists the instruction cycles, plus the ROMC state associated with each cycle, for every F8 instruction. Note that instructions are described in the table by order of ascending instruction (first byte) object code. *Table 4* lists the symbology used in *Table 3*.

Table 3 Instruction Cycle Execution and Timing

Op Code	Operand(s)	Object Code	Cycle	ROMC State	Timing	Status Flags				Interrupt	Function
						O	Z	C	S		
LR	A, KU	00	S	0	3S	—	—	—	—		A ← (r12)
LR	A, KL	01	S	0	3S	—	—	—	—		A ← (r13)
LR	A, QU	02	S	0	3S	—	—	—	—		A ← (r14)
LR	A, QL	03	S	0	3S	—	—	—	—		A ← (r15)
LR	KU, A	04	S	0	3S	—	—	—	—		r12 ← (A)
LR	KL, A	05	S	0	3S	—	—	—	—		r13 ← (A)
LR	QU, A	06	S	0	3S	—	—	—	—		r14 ← (A)
LR	QL, A	07	S	0	3S	—	—	—	—		r15 ← (A)

Table 3 Instruction Cycle Execution and Timing (Continued)

Op Code	Operand(s)	Object Code	Cycle	ROMC State	Timing	Status Flags O	Z	C	S	Interrupt	Function
LR	K, P	08	L	7	5	—	—	—	—		r12 ← (PC1U)
			L	B	5	—	—	—	—		r13 ← (PC1L)
			S	0	3S	—	—	—	—		
LR	P, K	09	L	15	2	—	—	—	—		PC1U ← (r12)
			L	18	2	—	—	—	—		PC1L ← (r13)
			S	0	3S	—	—	—	—		
LR	A, IS	0A	S	0	3S	—	—	—	—		A ← (ISAR)
LR	IS, A	0B	S	0	3S	—	—	—	—		ISAR ← (A)
PK		0C	L	12	2	—	—	—	—		PC1 ← (PC0);
			L	14	2	—	—	—	—		PC0L ← (r13)
			S	0	3S	—	—	—	—		PC0U ← (r12)
LR	P0, Q	0D	L	17	2	—	—	—	—	x	
			L	14	2	—	—	—	—		PC0L ← (r15)
			S	0	3S	—	—	—	—		PC0U ← (r14)
LR	Q, DC	0E	L	6	3	—	—	—	—		r14 ← (DC0U)
			L	9	5	—	—	—	—		r15 ← (DC0L)
			S	0	3S	—	—	—	—		
LR	DC,Q	0F	L	16	2	—	—	—	—		DC0U ← (r14)
			L	19	2	—	—	—	—		DC0L ← (r15)
			S	0	3S	—	—	—	—		
LR	DC, H	10	L	16	2	—	—	—	—		DC0U ← (r10)
			L	19	2	—	—	—	—		DC0L ← (r11)
			S	0	3S	—	—	—	—		
LR	H, DC	11	L	6	5	—	—	—	—		r10 ← (DC0U)
			L	9	5	—	—	—	—		r11 ← (DC0L)
			S	0	3S	—	—	—	—		
SR	1	12	S	0	3S	0	1/0	0	1		Shift (A) right one bit position (zero fill)
SL	1	13	S	0	3S	0	1/0	0	1/0		Shift (A) left one bit position (zero fill)
SR	4	14	S	0	3S	0	1/0	0	1		Shift (A) right four bit positions (zero fill)
SL	4	15	S	0	3S	0	1/0	0	1/0		Shift (A) left four bit positions (zero fill)
LM		16	L	2	6	—	—	—	—		A ← ((DC0))
			S	0	3S	—	—	—	—		
ST		17	L	5	1	—	—	—	—		(DC) ← (A)
			S	0	3S	—	—	—	—		
COM		18	S	0	3S	0	1/0	0	1/0		A ← (A) ⊕ H'FF' Complement accumulator
LNK		19	S	0	3S	1/0	1/0	1/0	1/0		A ← (A) + (C)
DI		1A	S	1C	0	—	—	—	—	y	Clear ICB
			S	0	3S	—	—	—	—		
EI		1B	S	1C	0	—	—	—	—		Set ICB
			S	0	3S	—	—	—	—	x	
POP		1C	S	4	0	—	—	—	—		PC0 ← (PC1)
			S	0	3S	—	—	—	—	x	

Table 3 Instruction Cycle Execution and Timing (Continued)

Op Code	Operand(s)	Object Code	Cycle	ROMC State	Timing	Status Flags O	Z	C	S	Interrupt	Function
LR	W, J	1D	S	1C	0	1/0	1/0	1/0	1/0		W ← (r9)
			S	0	3S	—	—	—	—	x	
LR	J, W	1E	S	0	3S	—	—	—	—		r9 ← (W)
INC		1F	S	0	3S	1/0	1/0	1/0	1/0		A ← (A) + 1
LI	aa	20	L	3	6	—	—	—	—		A ← H'aa'
	→	aa	S	0	3S	—	—	—	—		
NI	aa	21	L	3	4	0	1/0	0	1/0		A ← (A) v H'aa'
	→	aa	S	0	3S	—	—	—	—		
OI	aa	22	L	3	4	0	1/0	0	1/0		A ← (A) v H'aa'
	→	aa	S	0	3S	—	—	—	—		
XI	aa	23	L	3	4	0	1/0	0	1/0		A ← (A) ⊕ H'aa'
	→	aa	S	0	3S	—	—	—	—		
AI	aa	24	L	3	4	1/0	1/0	1/0	1/0		A ← (A) + H'aa'
	→	aa	S	0	3S	—	—	—	—		
CI	aa	25	L	3	4	—	—	—	—		Perform H'aa' + $\overline{(A)}$
	→	aa	S	0	3S	1/0	1/0	1/0	1/0		+ 1. Do not save result, but modify status flags to reflect result.
IN	PP	26	L	3	2	—	—	—	—		DB ← PP
	PP		L	1B	6	0	1/0	0	1/0		A ← (I/O Port PP)
			S	0	3S	—	—	—	—		
OUT	PP	27	L	3	2	—	—	—	—		DB ← PP
	PP		L	1A	1	—	—	—	—		I/O Port PP ← (A)
			S	0	3S	—	—	—	—	x	
PI	iijj	28	L	3	6	—	—	—	—		A ← H'ii'
	→	ii	S	D	0	—	—	—	—		PC1 ← (PC0) + 1
	→	jj	L	C	2	—	—	—	—		PC0L ← H'jj'
			L	14	1	—	—	—	—		PC0U ← (A)
			S	0	3S	—	—	—	—	x	
JMP	iijj	29	L	3	6	—	—	—	—		A ← H'ii'
	→	ii	L	C	2	—	—	—	—		PC0L ← H'jj'
	→	jj	L	14	1	—	—	—	—		PC0U ← (A)
			S	0	3S	—	—	—	—	x	
DCI	iijj	2A	L	11	2	—	—	—	—		DC0U ← ii
	→	ii	S	3	0						(increment PC0)
	→	jj	L	E	2	—	—	—	—		DC0L ← jj
			S	3	0						(increment PC0)
			S	0	3S	—	—	—	—		
NOP		2B	S	0	0	—	—	—	—		
XDC		2C	S	1D	0	—	—	—	—		DC0 ⇄ DC1
			S	0	0	—	—	—	—		
DS	r	3r	L	0	3L	1/0	1/0	1/0	1/0		r ← (r) + H'FF' Decrement scratchpad byte
LR	A, r	4r	S	0	3S	—	—	—	—		A ← (r)
LR	r, A	5r	S	0	3S	—	—	—	—		r ← (A)
LISU	e	6e	S	0	3S	—	—	—	—		ISARU ← 0'e'

Table 3 Instruction Cycle Execution and Timing (Continued)

Op Code	Operand(s)	Object Code	Cycle	ROMC State	Timing	Status Flags O	Z	C	S	Interrupt	Function
LISL	e	68 + e	S	0	3S	—	—	—	—		ISARL ← 0'e'
LIS	a	7a	S	0	3S	—	—	—	—		A ← H'0a'
BT	e, ii	8e	S	1C	0	—	—	—	—		Test e ∧ W register
		ii	S	3	0	—	—	—	—		Res = 0 so PC0 = (PC0) + 2
			S	0	3S						
			S	1C	0	—	—	—	—		
			L	1	2	—	—	—	—		Test e ∧ W register
						—	—	—	—		Res ≠ 0 so PC0 = (PC0)
			S	0	3S						+ H'ii' + 1
AM		88	L	2	4	—	—	—	—		
						1/0	1/0	1/0	1/0		A ← (A) + ((DC0)) Binary,
			S	0	3S						DC0 ← (DC) + 1
AMD		89	L	2	4	—	—	—	—		
						1/0	1/0	1/0	1/0		A ← (A) + ((DC0)) Decimal,
			S	0	3S	—	—	—	—		DC0 ← (DC0) + 1
NM		8A	L	2	4	0	1/0	0	1/0		A ← (A) ∧ ((DC0));
			S	0	3S	—	—	—	—		DC0 ← (DC0) + 1
OM		8B	L	2	4	0	1/0	0	1/0		A ← (A) ∧ ((DC0));
			S	0	3S	—	—	—	—		DC0 ← (DC0) + 1
XM		8C	L	2	4	0	1/0	0	1/0		A ← (A) ⊕ ((DC0));
			S	0	3S	—	—	—	—		DC0 ← (DC0) + 1
CM		8D	L	2	4	1/0	1/0	1/0	1/0		Set status flags on basis
			S	0	3S	—	—	—	—		of ((DC)) + (A) + 1;
						—	—	—	—		DC0 ← (DC0) + 1
ADC		8E	L	A	1	—	—	—	—		DC ← (DC) + (A)
			S	0	3S	—	—	—	—		
BR7	ii	8F	S	3	0	—	—	—	—		PC0 ← (PC0) + 2
		ii	S	0	3S	—	—	—	—		because (ISARL) = 7
			L	1	2	—	—	—	—		PC0 ← (PC0) + H'ii' + 1
			S	0	3S	—	—	—	—		because (ISARL) ≠ 7
BF	t, ii	9t	S	1C	0	—	—	—	—		Test t ∧ W. register
		ii	L	1	2	—	—	—	—		Res = 0 so PC0 = (PC0)
			S	0	3S	—	—	—	—		+ H'ii' + 1
			S	1C	0	—	—	—	—		Test t ∧ W. register
			S	3	0	—	—	—	—		Res ≠ 0 so PC0 = (PC0) + 2
			S	0	3S	—	—	—	—		
INS	0 or 1	A0, A1	S	1C	0	0	1/0	0	1/0		A ← (I/O Port 0 or 1)
			S	0	3S	—	—	—	—		
INS	4	A4	L	1C	0	0	1/0	0	1/0		DB ← Port address (4 thru 15)
	thru	thru	L	1B	6	—	—	—	—		
	15	AF	S	0	3S	—	—	—	—		A ← (Port 4 thru 15)
OUTS	0 or 1	B0, B1	S	1C	0	—	—	—	—		I/O Port 0 or 1 ← (A)
			S	0	3S	—	—	—	—		
OUTS	4	B4	L	1C	0	—	—	—	—		DB ← Port address (4 thru 15)
	thru	thru	L	1A	1	—	—	—	—		
	15	BF	S	0	3S	—	—	—	—	x	Port (4 thru 15) (A)

Table 3 Instruction Cycle Execution and Timing (Continued)

Op Code	Operand(s)	Object Code	Cycle	ROMC State	Timing	Status Flags O	Z	C	S	Interrupt	Function
AS	r	Cr	S	0	3S	1/0	1/0	1/0	1/0		A ← (A) + (r) Binary
ASD	r	Dr	S	1C	0	1/0	1/0	1/0	1/0		A ← (A) + (r) Decimal
			S	0	3S	—	—	—	—		
XS	r	Er	S	0	3S	0	1/0	0	1/0		A ← (A) ⊕ (r)
NS	r	Fr	S	0	3S	0	1/0	0	1/0		A ← (A) v (r)
INTRPT		xx	L	1C	0	—	—	—	—		IDLE
			L	0F	2	—	—	—	—		PC0L ← Int. address (lower byte); PC1 ← PC0
			L	13	2	—	—	—	—	y	PC0U ← Int. address (upper byte)
			S	0	3S	—	—	—	—	x	
RESET		xx	S	1C	0	—	—	—	—		IDLE
			L	8	1	—	—	—	—	y	PC0 ← 0, PC1 ← PC0
			S	0	3S	—	—	—	—	x	

Table 4 Instruction Execution and Timing Symbology

Symbol	Interpretation
A	The accumulator
$\overline{(A)}$	The complement of accumulator contents
a	A single hexadecimal digit being interpreted as data
aa	Two hexadecimal digits being interpreted as a single byte of data or as the high order byte of 16 bits of data
bb	Two hexadecimal digits being interpreted as the low order byte of 16 bits of data
Binary	Binary arithmetic specified
C	The carry status flag
DB	F8 system data bus
DC0	The primary data counter register
DC0L	The low order byte of the primary data counter register
DC0U	The high order byte of the primary data counter register
DC1	The secondary data counter register
Decimal	Decimal arithmetic specified
e	A single octal digit being interpreted as data
H	Scratchpad bytes 10 and 11
ii	Two hexadecimal digits being interpreted as the high order byte of a 16-bit address or as a simple byte address displacement

Symbol	Interpretation
ISAR	The 6-bit scratchpad address register
ISARL	The low order three bits of ISAR
ISARU	The high order three bits of ISAR
J	Scratchpad byte 9
jj	Two hexadecimal digits being interpreted as the low order byte of a 16-bit address
K	Scratchpad bytes 12 and 13
KL	Scratchpad byte 13
KU	Scratchpad byte 12
O	The overflow status flag
P	A single hexadecimal digit being interpreted as an I/O port address (0-15)
PP	Two hexadecimal digits being interpreted as an I/O port address (0-255)
PC0	The program counter register
PC0L	The low order byte of the program counter register
PC0U	The high order byte of the program counter register
PC1	The stack register
PC1L	The low order byte of the stack register
PC1U	The high order byte of the stack register
Q	Scratchpad bytes 14 and 15
QL	Scratchpad byte 15
QU	Scratchpad byte 14

Table 4 Instruction Execution and Timing Symbology (Continued)

Symbol	Interpretation
r	Single hexadecimal digit interpreted as scratchpad address: 4 = 0 through B for locations 0 through B in scratchpad r = C for ISAR as address source with no change after access r = D for ISAR as address source with ISARL = ISARL + 1 after access r = E for ISAR as address source with ISARL = ISARL−1 after access r = F is not allowed
S	The sign status flag
t	A single hexadecimal digit identifying a status condition that is tested by a Branch on Condition instruction
W	The status register

Symbol	Interpretation
Z	The zero status flag
∧	The logical OR of 8-bit quantities on each side of this symbol is specified
v	The logical AND of 8-bit quantities on each side of this symbol is specified
⊕	The logical Exclusive-OR of 8-bit quantities on each side of this symbol is specified
←	The value to the right of this symbol is to be loaded into the location specified on the left of this symbol
()	The contents of the location within the brackets is specified
(())	The contents of the memory word addressed by the contents of the location within the double brackets is specified
+	The binary address of 8-bit quantities on each side of this symbol is specified

ROMC Signal Functions

Table 5 describes the ROMC signals and their functions.

Table 5 ROMC Signal Functions

ROMC 4 3 2 1 0	HEX	Cycle Length	Function
0 0 0 0 0	00	S, L	Instruction Fetch. The device whose address space includes the contents of the PC0 register must place on the data bus the op code addressed by PC0; then all devices increment the contents of PC0.
0 0 0 0 1	01	L	The device whose address space includes the contents of the PC0 register must place on the data bus the contents of the memory location addressed by PC0; then all devices add the 8-bit value on the data bus, as a signed binary number, to PC0.
0 0 0 1 0	02	L	The device whose DC0 addresses a memory word within the address space of that device must place on the data bus the contents of the memory location addressed by DC0; then all devices increment DC0.
0 0 0 1 1	03	L, S	Similar to 00, except that it is used for Immediate Operand fetches (using PC0) instead of instruction fetches.
0 0 1 0 0	04	S	Copy the contents of PC1 into PC0.
0 0 1 0 1	05	L	Store the data bus contents into the memory location pointed to by DC0; increment DC0.
0 0 1 1 0	06	L	Place the high order byte of DC0 on the data bus.
0 0 1 1 1	07	L	Place the high order byte of PC1 on the data bus.
0 1 0 0 0	08	L	All devices copy the contents of PC0 into PC1. The CPU outputs zero on the data bus in this ROMC state. Load the data bus into both halves of PC0, thus clearing the register.
0 1 0 0 1	09	L	The device whose address space includes the contents of the DC0 register must place the low order byte of DC0 onto the data bus.
0 1 0 1 0	0A	L	All devices add the 8-bit value on the data bus, treated as a signed binary number, to the data counter.
0 1 0 1 1	0B	L	The device whose address space includes the value in PC1 must place the low order byte of PC1 on the data bus.

Table 5 ROMC Signal Functions (Continued)

ROMC 4 3 2 1 0	HEX	Cycle Length	Function
0 1 1 0 0	0C	L	The device whose address space includes the contents of the PC0 register must place the contents of the memory word addressed by PC0 onto the data bus; then all devices move the value that has just been placed on the data bus into the low order byte of PC0.
0 1 1 0 1	0D	S	All devices store in PC1 the current contents of PC0, incremented by 1; PC0 is unaltered.
0 1 1 1 0	0E	L	The device whose address space includes the contents of PC0 must place the contents of the word addressed by PC0 onto the data bus. The value on the data bus is then moved to the low order byte of DC0 by all devices.
0 1 1 1 1	0F	L	The interrupting device with highest priority must place the low order byte of the interrupt vector on the data bus. All devices must copy the contents of PC0 into PC1. All devices must move the contents of the data bus into the low order byte of PC0.
1 0 0 0 0	10	L	Inhibit any modification to the interrupt priority logic.
1 0 0 0 1	11	L	The device whose memory space includes the contents of PC0 must place the contents of the addressed memory word on the data bus. All devices must then move the contents of the data bus to the upper byte of DC0.
1 0 0 1 0	12	L	All devices copy the contents of PC0 into PC1. All devices then move the contents of the data bus into the low order byte of PC0.
1 0 0 1 1	13	L	The interrupting device with highest priority must move the high order half of the interrupt vector onto the data bus. All devices must move the contents of the data bus into the high order byte of PC0. The interrupting device resets its interrupt circuitry (so that it is no longer requesting CPU servicing and can respond to another interrupt).
1 0 1 0 0	14	L	All devices move the contents of the data bus into the high order byte of PC0.
1 0 1 0 1	15	L	All devices move the contents of the data bus into the high order byte of PC1.
1 0 1 1 0	16	L	All devices move the contents of the data bus into the high order byte of DC0.
1 0 1 1 1	17	L	All devices move the contents of the data bus into the low order byte of PC0.
1 1 0 0 0	18	L	All devices move the contents of the data bus into the low order byte of PC1.
1 1 0 0 1	19	L	All devices move the contents of the data bus into the low order byte of DC0.
1 1 0 1 0	1A	L	During the prior cycle, an I/O port timer or interrupt control register was addressed; the device containing the addressed port must move the current contents of the data bus into the addressed port.
1 1 0 1 1	1B	L	During the prior cycle, the data bus specified the address of an I/O port. The device containing the addressed I/O port must place the contents of the I/O port on the data bus. (Note that the contents of timer and interrupt control registers cannot be read back onto the data bus.)
1 1 1 0 0	1C	L or S	None.
1 1 1 0 1	1D	S	Devices with DC0 and DC1 registers must switch registers. Devices without a DC1 register perform no operation.
1 1 1 1 0	1E	L	The device whose address space includes the contents of PC0 must place the low order byte of PC0 onto the data bus.
1 1 1 1 1	1F	L	The device whose address space includes the contents of PC0 must place the high order byte of PC0 onto the data bus.

Timing Characteristics

The timing characteristics of the F3850 are described in *Table 6.* $V_{DD} = +5\ V \pm 5\%$, $V_{GG} = +12\ V \pm 5\%$, $V_{SS} = 0\ V$, $T_A = 0°C$ to $+70°C$

Table 6 F3850 CPU Signal Timing Characteristics

Symbol	Characteristic	Min	Typ	Max	Units	Test Conditions
P_x*	External Input Period	0.5		1.0	μs	
PW_x*	External Pulse Width	200		P_x - 200	ns	t_r, $t_f \leqslant 30$ ns
tx_1	Ext. to ϕ - to - Delay Extended Temp. Range			250 500	ns ns	$C_L = 100$ pF
tx_2	Ext. to ϕ + to + Delay Extended Temp. Range			250 500	ns ns	$C_L = 100$ pF
$P\phi$	ϕ Period	0.5		1.0	μs	
PW_1	ϕ Pulse Width	180		$P\phi$ - 180	ns	t_r, $t_f = 50$ ns; $C_L = 100$ pF
td_1	ϕ to WRITE + Delay Extended Temp. Range		150	250 400	ns ns	$C_L = 100$ pF
td_2	ϕ to WRITE - Delay Extended Temp. Range		150	250 400	ns ns	$C_L = 100$ pF
PW_2	WRITE Pulse Width	$P\phi$ - 100		$P\phi$	ns	t_r, t_f 50 ns typ; $C_L = 100$ pF
PW_S	WRITE Period; Short		$4P\phi$			
PW_L	WRITE Period; Long		$6P\phi$			
td_3	WRITE to ROMC Delay	80	300	550	ns	$C_L = 100$ pF
td_4*	WRITE to $\overline{ICB}$ Delay			350	ns	$C_L = 50$ pF
td_5	WRITE to $\overline{INT\ \ REQ}$ Delay			430	ns	$C_L = 100$ pF
t_{sx}*	$\overline{EXT\ \ RES}$ Setup Time	1.0			μs	$C_L = 20$ pF
t_{su}*	I/O Setup Time	300			ns	
t_h*	I/O Hold Time	50			ns	
t_o*	I/O Output Delay			2.5	μs	$C_L = 50$ pF
tdb_1*	WRITE to Data Bus Stable		0.6	1.3	μs	$C_L = 100$ pF
tdb_2	WRITE to Data Bus Stable	$2P\phi$		$2P\phi + 1.0$	μs	$C_L = 100$ pF
tdb_3*	Data Bus Setup	200			ns	
tdb_4*	Data Bus Setup	500			ns	
tdb_5	Data Bus Setup	500			ns	
tdb_6*	Data Bus Setup	500			ns	

1. Symbols marked with an asterisk (*) refer to parameters that are most frequently of importance when interfacing to an F8 system. They encompass I/O timing, external timing generation, and possible external RAM timing. The remaining parameters are typically those that are only relevant between F8 devices, and not normally of concern to the user.
2. Input and output capacitance is 3 to 5 pF typical on all pins except V_{DD}, V_{GG}, and V_{SS}.
3. If $\overline{INT\ \ REQ}$ is being supplied asynchronously, it can be pulled down at any time except during a fetch cycle that has been preceded by a non-privileged instruction. In that case $\overline{INT\ \ REQ}$ must go down according to the requirements of td_5.

DC Characteristics

The DC characteristics of the F3850 are provided in *Table 7*. $V_{DD} = +5\,V \pm 5\%$, $V_{GG} = +12\,V \pm 5\%$, $V_{SS} = 0V$, $T_A = 0°C$ to $+70°C$

Table 7 F3850 CPU Signal DC Characteristics

Signal	Symbol	Characteristic	Min	Max	Unit	Test Conditions
ϕ, WRITE	V_{OH}	Output High Voltage	4.4	V_{DD}	V	$I_{OH} = -50\,\mu A$
	V_{OL}	Output Low Voltage	V_{SS}	0.4	V	$I_{OL} = 1.6\,mA$
	V_{OH}	Output High Voltage	2.9		V	$I_{OH} = -100\,\mu A$
XTLY	V_{IH}	Input High Voltage	4.5	V_{GG}	V	
	V_{iL}	Input Low Voltage	V_{SS}	0.8	V	
	I_{IH}	Input High Current	5	50	μA	$V_{IN} = V_{DD}$
	I_{IL}	Input Low Current	−10	−120	μA	$V_{IN} = V_{SS}$
$ROMC_{0-4}$	V_{OH}	Output High Voltage	3.9	V_{DD}	V	$I_{OH} = -100\,\mu A$
	V_{OL}	Output Low Voltage	V_{SS}	0.4	V	$I_{OL} = 1.6\,mA$
DB_{0-7}	V_{IH}	Input High Voltage	2.9	V_{DD}	V	
	V_{IL}	Input Low Voltage	V_{SS}	0.8	V	
	V_{OH}	Output High Voltage	3.9	V_{DD}	V	$I_{OH} = -100\,\mu A$
	V_{OL}	Output Low Voltage	V_{SS}	0.4	V	$I_{OL} = 1.6\,mA$
	I_{IH}	Input High Current		3	μA	$V_{IN} = 7\,V$ 3-State mode
	I_{IL}	Input Low Current		−3	μA	$V_{IN} = V_{SS}$ 3-State mode
I/O_{0-17}	V_{OH}	Output High Voltage	3.9	V_{DD}	V	$I_{OH} = -30\,\mu A$
	V_{OH}	Output High Voltage	2.9	V_{DD}	V	$I_{OH} = -150\,\mu A$
	V_{OL}	Output Low Voltage	V_{SS}	0.4	V	$I_{OL} = 1.6\,mA$
	V_{IH}	Input High Voltage(1)	2.9	V_{DD}	V	Internal pull-up to V_{DD}
	V_{IL}	Input Low Voltage	V_{SS}	0.8	V	
	I_{IL}	Input Low Current		−1.6(4)	mA	$V_{IN} = 0.4\,V$(2)
$\overline{EXT\ RES}$	V_{IH}	Input High Voltage	3.5	V_{DD}	V	Internal pull-up to V_{DD}
	V_{IL}	Input Low Voltage	V_{SS}	0.8	V	
	I_{IL}	Input Low Current	−0.1	−1.0	mA	$V_{IN} = V_{SS}$
$\overline{INT\ REQ}$	V_{IH}	Input High Voltage	3.5	V_{DD}	V	Internal pull-up to V_{DD}
	V_{IL}	Input Low Voltage	V_{SS}	0.8	V	
	I_{IL}	Input Low Current	−0.1	−1.0	mA	$V_{IN} = V_{SS}$
$\overline{ICB}$	V_{OH}	Output High Voltage	3.9	V_{DD}	V	$I_{OH} = -10\,\mu A$
	V_{OH}	Output High Voltage	2.9	V_{DD}	V	$I_{OH} = -100\,\mu A$
	V_{OL}	Output Low Voltage	V_{SS}	0.4	V	$I_{OL} = 100\,\mu A$

1. Hysteresis input circuit provides additional 0.3 V noise immunity while internal pull-up provides TTL compatibility.
2. Measured while F8 port is outputting a high level.
3. Guaranteed but not tested.
4. −1.8 V max. for extended temperature range.
5. Positive current is defined as conventional current flowing into the pin referenced.

Supply Currents

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
I_{DD}	V_{DD} Current		45	75	mA	f = 2 MHz, Outputs Unloaded
I_{GG}	V_{GG} Current		12	30	mA	f = 2 MHz, Outputs Unloaded

Recommended Operating Ranges

The recommended operating ranges of the F3850 are shown below.

Absolute Maximum Ratings

V_{GG}	−0.3 V, +15 V
V_{DD}	−0.3 V, +7 V
XTLX, XTLY, and XTLZ	−0.3 V, +15 V
All other inputs	−0.3 V, +7 V
Storage temperature	−55°C, +150°C
Operating temperature	0° C, +70°C

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

	Supply Voltage (V_{DD})			Supply Voltage (V_{GG})			
Part Number	Min	Typ	Max	Min	Typ	Max	V_{SS}
F3850	+4.75 V	+5 V	+5.25 V	+11.4 V	+12 V	+12.6 V	0 V

Ordering Information

Order Code	Package	Temperature Range
F3850DC	Ceramic	0°C to +70°C
F3850DM	Ceramic	−55°C to +125°C
F3850PC	Plastic	0°C to +70°C

FAIRCHILD
A Schlumberger Company

F3851/F3856
Program Storage Unit

Microprocessor Product

3

Description

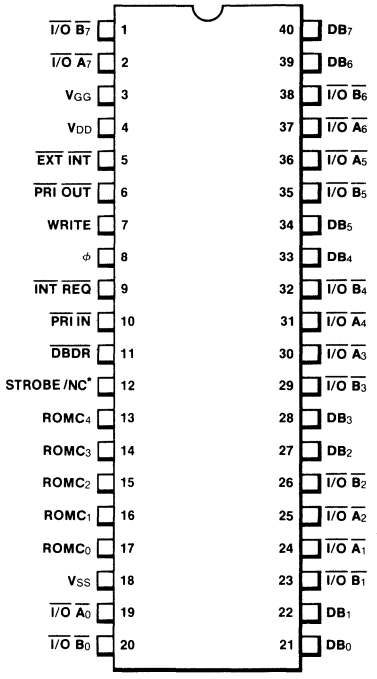
The Fairchild F3851 and F3856 are the principal program storage devices for the F8 microcomputer system. The F3851 provides 1024 bytes of ROM; the F3856 provides 2048 bytes. The program storage unit (PSU) is customized with programs and permanent data tables, which are specified as ROM masks.

The PSU devices have two 8-bit, bidirectional I/O ports, interrupt logic, a programmable timer, and a pulse width measurement circuit. They also contain memory addressing logic with data counters and program counters. The interrupt logic responds to requests from an external device and internally from the timer. The pulse width measurement circuit (F3856) is a combination of these two capabilities.

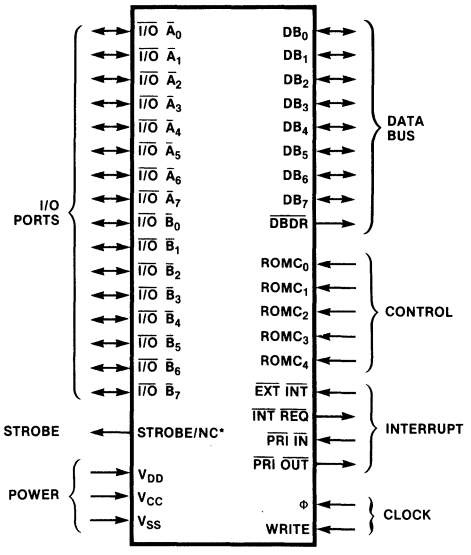
The PSU devices are manufactured using N-channel, isoplanar MOS technology; therefore, power dissipation is very low, typically less than 275 mW.

- **1024/2048 Bytes of Program Storage**
- **Internal Memory Addressing Logic**
- **16 Bidirectional, Individually Controlled I/O Lines, Organized as Two 8-Bit Ports**
- **Programmable Timer (F3856) — Preset, Start, Stop, and Read-Back Ability; Four Selectable Timer Count Rates, and Pulse Width Measurement**
- **Full Interrupt Level — Daisy-Chain Expandable, Independent Interrupt Address Vectors for Timer and External Interrupt**
- **2 MHz Operation**
- **TTL and LSTTL Compatible**
- **Low Power Dissipation, Typically Less Than 275 mW**
- **+5 V and +12 V Power Supplies**

Connection Diagram



Signal Functions



*NC for F3851 only.

Device Organization

The PSU is more than a read-only memory unit: every memory device within the F8 system contains its own memory addressing logic along with associated address registers. Refer to figure 1 for a simplified block diagram of the PSU. A single 8-bit data bus provides all necessary communication between a PSU (or any other memory device) and an F3850 CPU.

The PSU has an elementary arithmetic unit that can increment and add 16-bit data units; for memory addressing logic, these two operations are sufficient. The PSU is functionally illustrated in figure 2. These devices also contain a control unit that decodes the five read-only memory control (ROMC) lines, generated by the CPU, as though they were a 5-bit instruction code. Similar to the CPU, the PSU generates internal signals to control data flow and arithmetic logic within itself. One control output, data bus drive (DBDR), is generated to coincide with data being output by the PSU.

System Clock Timing

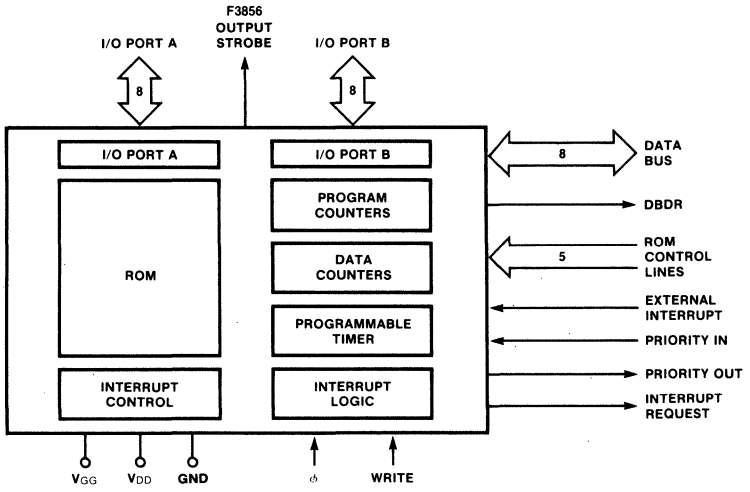
All timing within the F3851/F3856 PSU is controlled by the ϕ and WRITE signals, which are generated from the F3850 CPU. Refer to the F3850 data sheet for a description of these clock signals, The WRITE clock refreshes and updates PSU address registers, which are dynamic. The ϕ clock drives sequencing logic to precharge the ROM matrix; it also drives the programmable timer.

I/O Ports

The unit contains four preassigned I/O port addresses: the two lowest are assigned to I/O ports A and B and are used to transfer data to and from external devices. The other two I/O addresses are assigned to the programmable timer and the interrupt control register and are treated as I/O ports. Associated with the I/O ports is an I/O port address select register (ASR). This is a 6-bit register for the F3851 and a 5-bit register for the F3856. The contents are a mask option, which must be specified at the time the PSU is created. The ports are addressed as follows:

XXXXXX00	I/O port A
XXXXXX01	I/O port B
XXXXXX10	Interrupt control register
XXXXXX11	Programmable timer

Figure 1 PSU Simplified Block Diagram



For example, if the six binary digits are 000010, the four I/O port addresses are H'08', H'09', H'0A', and H'0B'.

When a logic 1 is output to I/O port A or B, it places a 0 V level on the output pin. This same inverted logic applies to input.

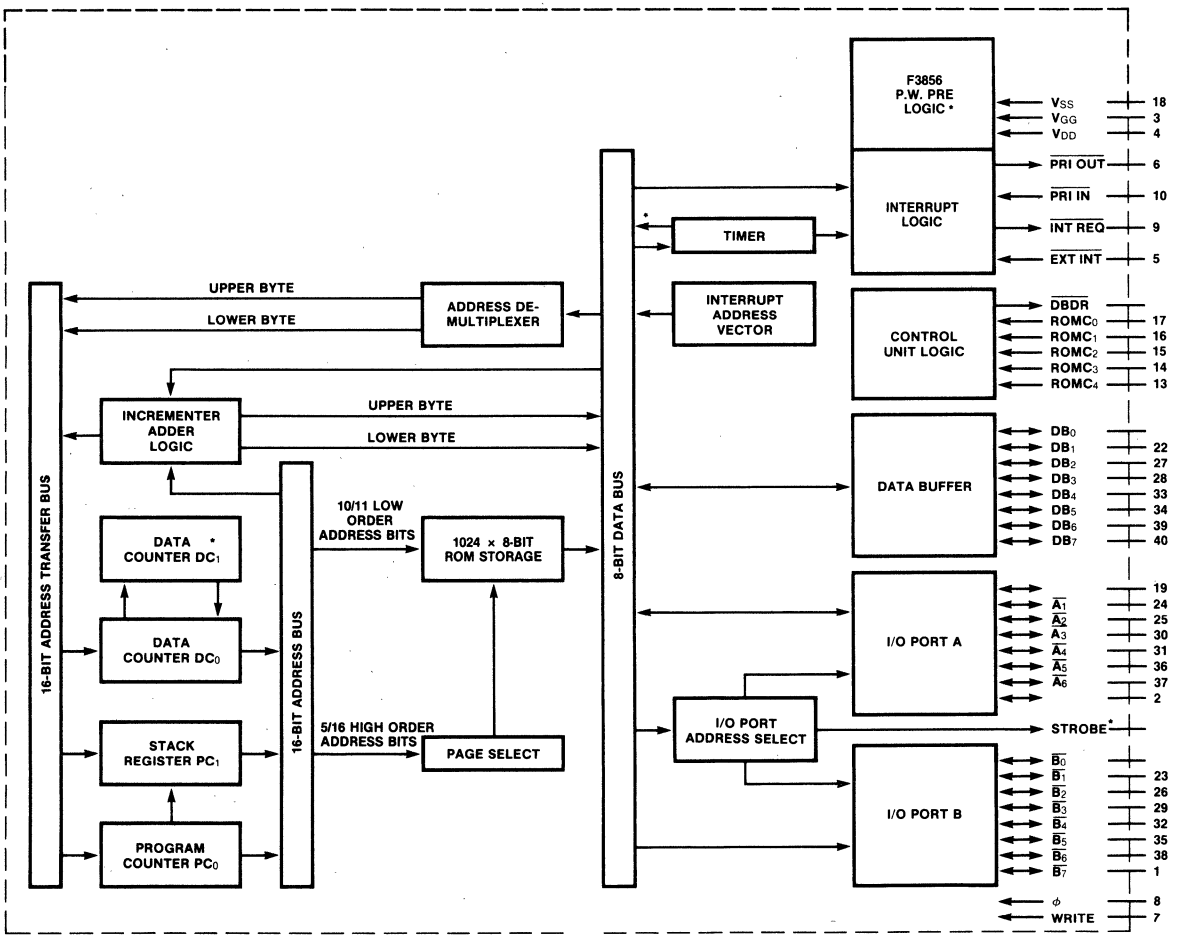
The F3851 I/O ports, timer, and interrupt control register are not initialized during the power-on reset cycle. The F3856 I/O ports and interrupt control register are initialized during both the power-on or external reset cycle; the timer register is not initialized during power-on or external reset cycles.

ROM Addressing

The F3851 8K PSU has 1024 bytes of read-only memory; the F3856 16K PSU has 2048 bytes. This ROM array may contain object program code and/or tables of nonvarying data. Every PSU is implemented using a custom mask that specifies the state of every ROM bit and certain address mask options that are external to the ROM array.

3

Figure 2 PSU Functional Diagram



The ROM addressing logic consists of 16-bit registers: program counter PC_0, stack register PC_1, and data counter DC_0. Data counter DC_1 is provided on the F3856 as an additional buffer for DC_0.

A 6-bit page select register and 10-bit address select register provide decode logic for the F3851. The F3856 uses a 5-bit page select register and an 11-bit address select register.

Program Counter, Data Counter, and Stack Registers
Program counter PC_0 always addresses the memory location out of which the next program instruction byte is read. If the instruction requires data (i.e., an operand) to be accessed, data counter DC_0 must address memory for this purpose; PC_0 cannot be used to address data, since it is saving the address of the next instruction code. By using the exchange DC instruction in the F3856 program, the two data counter contents of DC_0 and DC_1 can be exchanged.

The provision of two address registers, PC_0 and DC_0, is a convenience to the F3850 CPU and is not a necessary part of the memory addressing logic sequence within a PSU. Address decoding is identical, whether originating in PC_0 or DC_0.

The PC_0, PC_1, and DC_0 are loaded from two consecutive single-byte inputs on the data bus; PC_1 and DC_0 are transmitted as two single-byte outputs on the data bus. The contents of DC_0 and DC_1 of F3856 can be exchanged in one instruction.

Stack register PC_1 is a buffer for program counter PC_0; the contents of PC_1 are never used directly to address memory. When an interrupt is acknowledged, the contents of PC_0 are saved in PC_1.

Page Select and Address Select Registers
All memory addresses are 16 bits wide, whether originating in the program counter or in the data counter. Address decode logic within the PSU separates the 16-bit address into two portions: the low order addresses the ROM storage bytes; the high order addresses the page.

	High-Order Byte Address	Low-Order Page Address
F3851	1024 Byte Select 6 Bits	64 Page Options 10 Bits
F3856	2048 Byte Select 5 Bits	32 Page Options 11 Bits

If the high-order bits of the address coincide exactly with the page select mask, an enable signal is generated, causing the PSU logic to respond to a memory access request. If the high-order bits of the address do not coincide exactly with the page select, no enabling signal is generated and the PSU does not respond to memory access requests.

The page select register identifies the memory addressing space of the individual PSU device. Each of the 32 (or 64) page select options allowed by the 5-bit (or 6-bit) page select register identifies a single address space consisting of 2048 (or 1024) continguous memory addresses.

Incrementer Adder Logic
There are only two arithmetic operations that memory devices need to perform on the contents of memory address registers:

1. Increment by 1 the 16-bit value stored in address PC_0 or DC_0.

2. Add an 8-bit value, treated as a signed binary number (subject to twos complement arithmetic) to the 16-bit value stored in an address register. If the 8-bit value is being treated as a signed binary number, the high-order bit of the 8-bit value is the sign bit; the sign bit must be propagated through the missing high-order eight bits.

The PSU control unit implements the incrementer adder logic through control signals internal to PSU device logic.

Addressing Consistency in Multiple Memory Devices
When an ROMC state specifies a memory access, only one memory device responds to the memory access operation itself. However, every memory device responds to ROMC states that call for modifying the contents of a program counter or data counter register. Providing every memory device that is connected to the 8-bit data bus of an F3850 CPU is also connected to the ROMC control lines of the same CPU, address contentions cannot arise. Every memory device simultaneously receives the same ROMC state signals from the CPU; every memory device responds to ROMC states by identically modifying the contents of memory address registers, if such modifications are specified. Therefore, every PC_0 register on every memory device always contains identical information; the same is true for DC_0 and PC_1 registers.

Only one memory device (the one whose address space includes the specified memory address) actually responds to any memory access request. To avoid addressing conflicts, it is only necessary to ensure that the following conditions exist:

Signal Descriptions
The PSU input and output signals are described in table 1.

Table 1 PSU Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
ϕ	8	Clock	The two clock input signals that originate at the F3850 CPU.
WRITE	7		
I/O Ports			
$\overline{I/O\ A_0}$-$\overline{I/O\ A_7}$	19, 24, 25, 30, 31, 36, 37, 2	I/O Ports A	Bidirectional ports through which the PIO communicates with logic external to the microprocessor system.
$\overline{I/O\ B_0}$-$\overline{I/O\ B_7}$	20, 23, 26, 29, 31, 35, 38, 1	I/O Ports B	
Control			
$ROMC_0$-$ROMC_4$	17, 16, 15, 14, 13	Read-Only Memory Control	Input signals that originate at the F3850 CPU control internal functions of the PSU.
Data Bus			
DB_0-DB_7	21, 22, 27, 28, 33, 34, 39, 40	Data Bus	Bidirectional 3-state lines that link the PSU to all other devices within the microprocessor system.
$\overline{DBDR}$	11	Data Bus Drive	A low output, open-drain signal that indicates the data bus currently contains data flowing from the PSU.
Strobe			
STROBE	12	Strobe	This output signal provides a positive pulse whenI/O port A is being read by an input instruction or is being updated by an output instruction (F3856).
Interrupt			
$\overline{EXT\ INT}$	5	External Interrupt	A high-to-low transition on this input signal is interpreted as an interrupt request from an external device.
$\overline{INT\ REQ}$	9	Interrupt Request	This output signal is the $\overline{INT\ REQ}$ input to the F3850 CPU; it must be output low to interrupt the CPU, which occurs only if $\overline{PRI\ IN}$ is low and PSU interrupt control logic is requesting an interrupt.
$\overline{PRI\ IN}$	10	Priority In	Unless this input signal is low, the PSU does not set the $\overline{INT\ REQ}$ signal low in response to an interrupt.
$\overline{PRI\ OUT}$	6	Priority Out	This output signal becomes the $\overline{PRI\ IN}$ signal to the next device in the interrupt-priority daisy chain; it is output high unless the $\overline{PRI\ IN}$ signal is entering the PSU low and the PSU is not requesting an interrupt
Power			
V_{DD}	4	Power Supply	+5 V ±5%
V_{GG}	3	Power Supply	+12 V ±5%
V_{SS}	18	Ground	System ground — 0 V; V_{DD} and V_{GG} are referenced to V_{SS}.

1. All memory devices must receive the same ROMC state signals from one CPU and must contain identical information.

2. Page select masks must not be duplicated — more than one memory device cannot have the same memory space.

3. The memory address contained in the specified register (PC_0 or DC_0) must be within the memory space of at least one memory device.

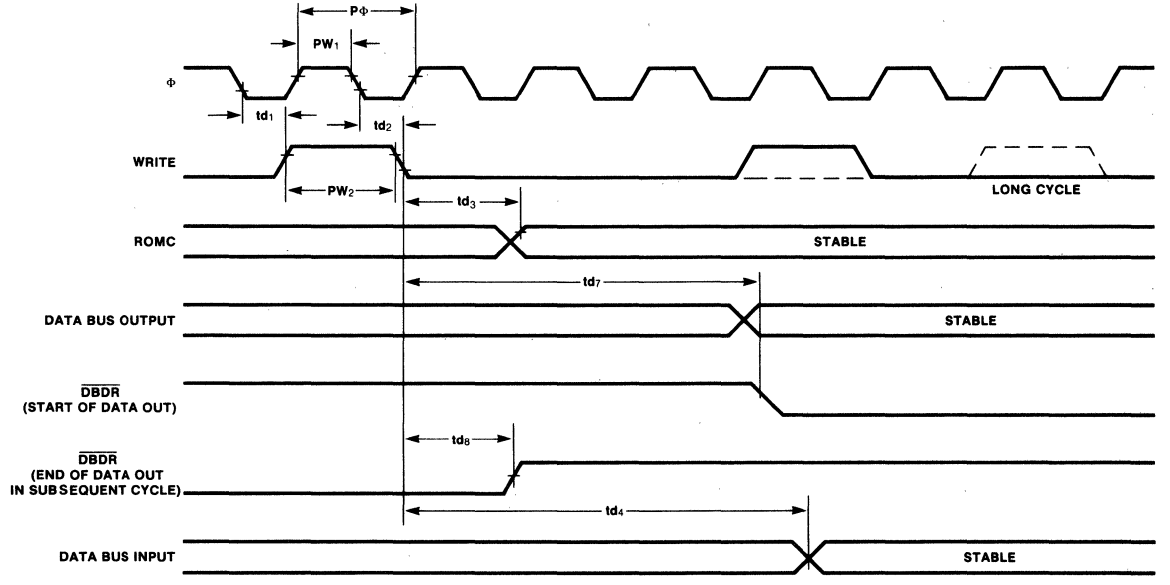
ROMC States
Table 2 lists the data bus contents as a function of ROMC states.

Instruction Execution

The PSU responds to signals that are output by the F3850 CPU in the course of implementing instruction cycles. Refer to table 2 for a summary of the data bus response to the ROMC states generated by the CPU.

Data Output by the PSU
Figure 3 provides timing when the PSU outputs data on the

Figure 3 Data Bus Timing



data bus. This timing applies whenever a PSU is the data source. The PSU places data on the data bus, even in the worst case, in time for the setup required by any F3850 CPU destination (refer to the F3850 CPU data sheet).

Table 2 Data Bus Contents as a Function of the ROMC State

ROMC State (Hex)	If F3851/F3856 PSU is the Source*		If F3850 CPU is the Source
	Description of Data	Address**	Description of Data
00	Instruction	PC_0	
01	Offset for branch	PC_0	
02	Operand	DC_0	
03	Operand	PC_0	
04			
05			Byte to be stored
06	Upper byte, DC_0		
07	Upper byte, PC_1		
08			= 00 for PC_0
09	Lower byte, DC_0		
0A			Offset for DC_0
0B	Lower byte, PC_1		
0C	Byte for PC_0, lower	PC_0	
0D			
0E	Byte for DC_0, lower	PC_0	
0F	Lower byte of interrupt vector if it is source of the interrupt		
10			
11	Byte for DC_0, upper	PC_0	
12			Byte for PC_0, lower
13	Upper byte of interrupt vector if it is source of the interrupt		
14			Byte for PC_0, upper
15			Byte for PC_1, upper
16			Byte for DC_0, upper
17			Byte for PC_0, lower
18			Byte for PC_1, lower
19			Byte for DC_0, lower
1A			Byte for selected I/O port
1B	Byte from I/O register, if selected		
1C			(Note 1)
1D			
1E	Lower byte, PC_0		
1F	Upper byte, PC_0		

*Only drives the data bus within the segment of address space that belongs to the PSU.
**An entry in this column specifies the register from which a memory address was obtained.

Note 1
During INS or OUTS instruction for port 0 or 1: I/O byte
During INS or OUTS instruction for port 4-F: I/O address
During all other instructions, F3850 does not drive.

The data bus drive signal ($\overline{DBDR}$) is low, while data output by the PSU is stable on the data bus. Thus, a $\overline{DBDR}$ low signal indicates that the data bus currently contains data flowing from a PSU. For systems with more than one PSU, the $\overline{DBDR}$ outputs can be wire-ORed and the result used as a bus data flow direction indicator. The $\overline{DBDR}$ signal remains low until timing delay td_8 into the instruction cycle following the one in which $\overline{DBDR}$ was set low.

Data Input to the PSU

When the PSU receives data off the data bus, in the worst case, the data must be added to a 16-bit number within the PSU adder/incrementer. This worst case corresponds to data coming from the accumulator of the CPU for an ADC instruction or from a memory device for a BR instruction. For this worst case, arriving data must allow sufficient time for 16-bit adder logic (time delay td_4 in figure 3 identifies this worst-case timing).

PSU Input/Output Interfacing

The I/O ports with addresses XXXXXX00 and XXXXXX01 (XXXXXX is the 6-bit I/O port address select) are used to transmit data between the PSU and external devices. The IN and INS instructions cause data at the I/O ports to be transmitted to the CPU; the OUT and OUTS instructions cause data in the CPU accumulator to be loaded into an I/O port. Each I/O pin has an output latch that holds the pin DC data.

Input and output operations using the two PSU I/O ports execute in three instruction cycles. During the first cycle, the port address is transmitted to the data bus. During the second cycle, data is either sent from the accumulator to the I/O latch or enabled from the I/O pin to the accumulator, depending on whether the instruction is an output or an input. At the falling edge of the WRITE signal (marking the end of the second cycle and beginning of the third cycle), the data is strobed into either the latch (OUTS) or the accumulator (INS), respectively. The third cycle is then used by the CPU for its next instruction fetch.

I/O Port Options

Data bus timing associated with the execution of I/O instructions does not differ from data bus timing associated with any other data transfer to or from the PSU. However, timing at the I/O port itself depends on which port option is being used. Figures 4, 5, and 6 illustrate the three port options; figure 7 illustrates timing for the three cases.

Standard Pull-Up Configuration (Figure 4)—All I/O port bits should be set for a high level, before data input, to prevent incoming logic 0s from being masked by logic 1s preset at the port from previous outputs. In some instances, the ability to mask bits of a port to logic 1 is useful. (Note that logic 1 becomes a 0 V electrical level at the I/O pin; logic 0 corresponds to a high electrical level.)

Each I/O port pin is a wire-AND structure between an internal latch and an external signal, if any. The latch is always loaded directly from the accumulator. Each I/O pin is set high or low under program control. If a 1 (high) is presented at the latch, gate (b) turns on and gate (a) turns off, so that P is at V_{SS} (low). If a 0 (low) is presented at the latch, gate (a) turns on and gate (b) turns off, so that P is at V_{DD} (high).

When data is output through an I/O port, the pin is connected directly to a standard TTL gate input. Data is input to the pin from a TTL output. In normal operation, high or low levels at P drive the external TTL device input transistor (d). If a low level is set at P, transistor (d) conducts current through the path J, I, P, and FET (b). This is transferred as a low level to the rest of the circuits in the TTL device and results in a high or low level at the output of the device, depending on its characteristics. If the level at P is set high, transistor (d) cuts off and a high level is transferred by (d). When data is input to the I/O pin, a high or low signal at the pin transfers a logic 1 or 0 to the accumulator.

Since the I/O pin and the TTL device output at 0 are wire-ANDed, it is possible for the state of one to affect the transfer of data out from the I/O pin or in from the TTL device output. For example, if the latch in the I/O port is set so that the pin is clamped low by (b), the level at 0 cannot pull P high. Conversely, if P is clamped to a low level by (c), setting the latch for a high level has no effect.

Open-Drain Configuration (Figure 5)—When the I/O port is configured as shown in figure 5, the drain connection of FET (a) is open, i.e., not connected to V_{DD} through a pull-up transistor. This option is most useful in applications where several signals (possibly several I/O port lines) are to be wire-ORed together. A common external pull-up, R_L, is used to establish the logic 1 levels. Another advantage of this option is that the output (point Y) can be tied through a pull-up resistor to a voltage higher than V_{DD} (clear up to V_{GG}) for interfacing to external circuits requiring a higher logic 1 level than V_{DD} provides.

If a high level is present at point X (coming from the port latch), FET (a) will conduct and pull point Y to a low level by current flow through R_L. This low level at Y causes transistor (b) to turn on and present a low level to the input TTL circuit.

If a low level is present at X, FET (a) turns off and point Y is pulled toward V_{DD} by R_L. This causes transistor (b) to turn off and present a high level to the internal TTL circuits.

When data is input, a high level at the base of transistor (c) causes (c) to conduct and pull point Y low, with current flow through T_L. This transfers a high level to the internal I/O port logic through inverting action by the hysteresis circuit. If a

Figure 4 Standard Pull-Up Configuration

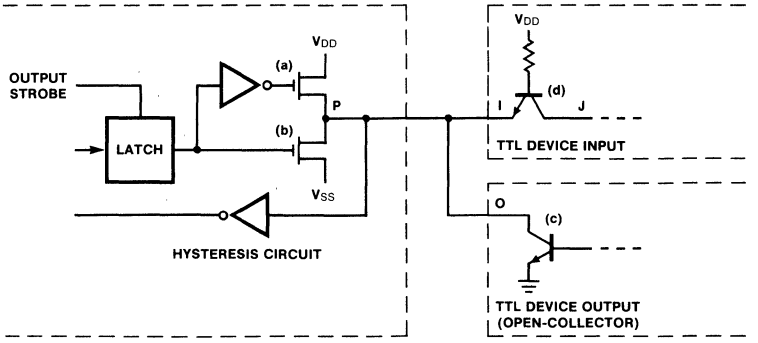


Figure 5 Open-Drain Configuration

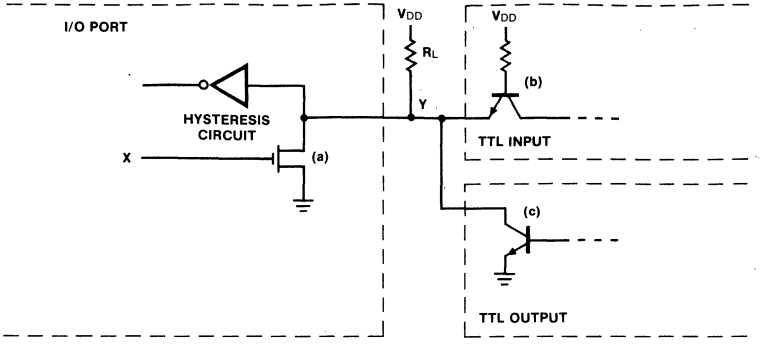
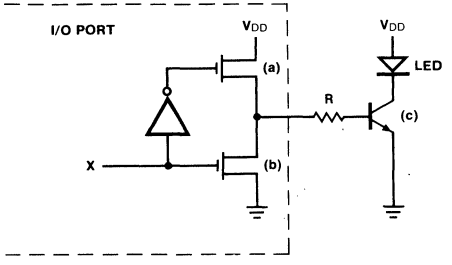


Figure 6 Driver Pull-Up Configuration



low level is present at the base of (c), conduction stops and point Y is pulled toward V_{DD} by R_L. This is then transferred as a low level to internal I/O port logic through the hysteresis circuit.

Driver Pull-Up Configuration (Figure 6)— Figure 6 shows the I/O port driver pull-up option used to drive an LED indicator. This application is typical of a front-panel address or data display, where a row of LED indicators shows the logic state at each pin of an I/O port.

A high level at X turns FET (b) on and (a) off, providing a path for current through resistor R from the base of transistor (c). This stops (c) from conducting and the LED does not light. If a low level is present at X, (b) turns off and (a) turns

on, providing a path for current from V_{DD} through (a) to R. This current through R turns on (c), which causes the LED to conduct and be lighted.

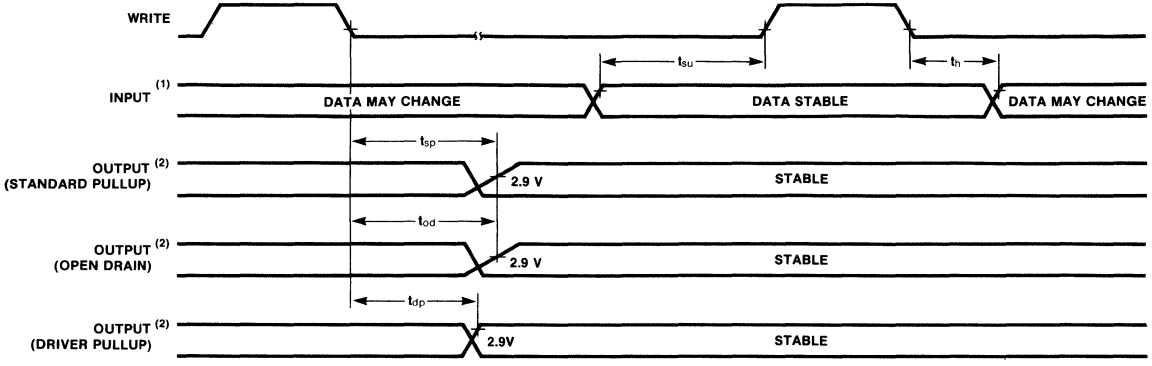
The three options for I/O port output configurations described above are provided to aid the designer in optimizing (minimizing) the system hardware for a particular application. The choice in configuration is specified as a mask option by the designer.

During input instrucions, the trailing edge of the STROBE signal is used to indicate to the external device that the current data on the I/O port is read and new data can be changed. For example, if a shift register is connected to the I/O port, the trailing edge of the STROBE signal is used to advance the shift register.

During output instruction, the trailing edge of this STROBE signal indicates that the new data on the I/O port latches is being changed. The output on the latches becomes true after typically 500 ns of the trailing edge of this signal.

Figure 7 PSU I/O Port Timing

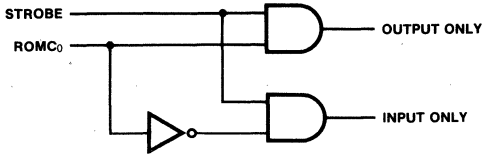
Refer to the "Timing Characteristics" section for all signal characteristics.



F3856 I/O Strobe

An additional output strobe signal is provided on the F3856 to indicate the execution of an input or output instruction for the low address I/O port on the PSU circuit. (This is port 4 of the PSU circuit with the 4-7 address.) A pulse of the duration of the WRITE clock on the STROBE pin is provided at the end of the second cycle of the I/O instruction for this port. Figure 8 shows the timing relationship of this output with respect to the execution being performed.

Although this pulse appears for both input and output instructions for this port, two different signals for input only are derived from the external gating of the STROBE and $ROMC_0$ signals, as shown below.



F3851 Programmable Timer

The F3851 PSU has an 8-bit shift register, addressable as I/O port XXXXXX11, that can be used as a programmable timer (XXXXXX is the 6-bit I/O port address select, a PSU mask option). Figure 9 illustrates the shift register logic and the exclusive-OR feedback path.

Based on the logic illustrated in figure 9, binary values in the range 0 through 254, when loaded into the timer, are converted into "timer counts." As shown in table 3, "timer contents" is the actual binary value loaded into a timer, and "timer counts" is the corresponding number of time intervals the timer takes to time out. Data cannot be read out of the programmable timer I/O port.

As described in the *Guide to Programming the F8 Microcomputer,* an assembly-language program specifies timer counts, and the assembler converts timer counts into the binary value that must be loaded into the programmable timer. This is the value given under "Contents" in table 3. To

Figure 8 I/O Instruction Fetch and Strobe Timing

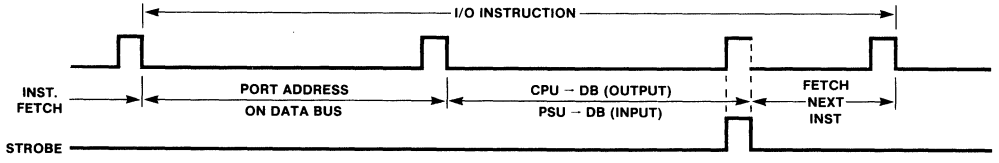
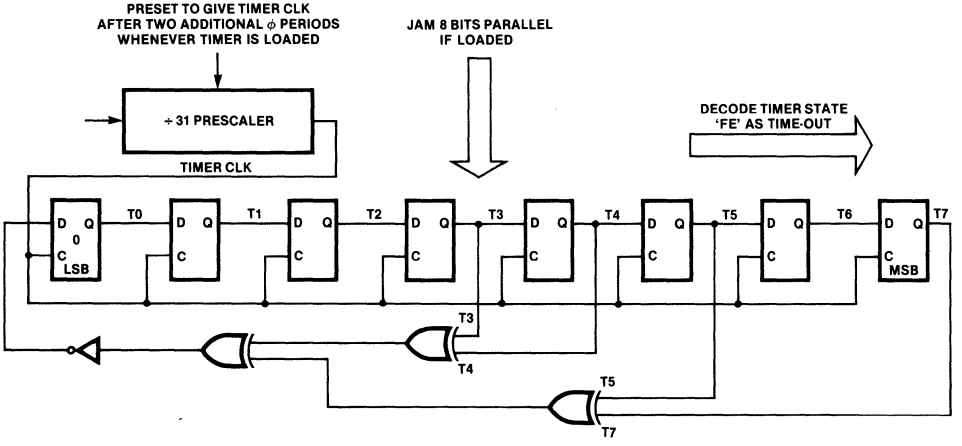


Figure 9 F3851 Timer Block Diagram



use a programmable timer, bypassing assembly-language programming, load the programmable timer with the value given under "Contents" in table 3 to time out after the number of intervals given under "Counts."

It is also possible to write small subroutines that calculate time values one count faster or slower than a given value. Such subroutines would be used if programmed delays are required.

The OUT or OUTS instruction is used to load timer counts into the programmable timer. The contents of the programmable timer cannot be read using an IN or INS instruction. The timer times out after a time interval given by the product (period of ϕ clock × (timer counts) × 31). For example, a value of 200 (11001000, or H'C8') loaded into the programmable timer becomes 215 timer counts. The timer, therefore, times out in 3.33 ms, if the period of clock signal ϕ is 500 ns.

A value of 255 (H'FF') loaded into a programmable timer stops the timer.

All timers run continuously, unless they have been stopped by loading H'FF' into the timer. Upon timing out, the timer transmits an interrupt request to the interrupt logic. If proper interrupt logic conditions exist, the timer interrupt request is passed on to the CPU through the $\overline{\text{INT REQ}}$ signal.

After a programmable timer has timed out, it again times out after 255 timer counts; therefore, if the programmable timer is left running, it times out every 7905 ϕ clock periods, or every 3.953 ms for a 500 ns clock.

If the timer is actually loaded with a zero value, it times out in 24 counts, whereas, once it has timed out, it next times out in 255 counts; i.e., a time-out is not the same thing as counting down to zero.

When the timer and timer interrupt are being set to time a new interval, the timer is always loaded before enabling the timer interrupt. Loading the timer clears any pending timer interrupts. When the timer interrupt is enabled, any pending

timer interrupt is acknowledged and forwarded to the CPU. Since the timer runs continuously, unless stopped under program control, enabling the timer before loading a time count can cause errors. Prior time-outs of the timer are latched in the interrupt logic of the PSU, even while timer interrupts are disabled. When the timer is enabled, an immediate interrupt acknowledge occurs if, by chance, the continuous-running timer happens to time out while timer interrupts are disabled.

If the timer is loaded just before enabling timer interrupts, loading the timer clears pending timer interrupts. Now a spurious interrupt request does not exist when the timer interrupt is enabled.

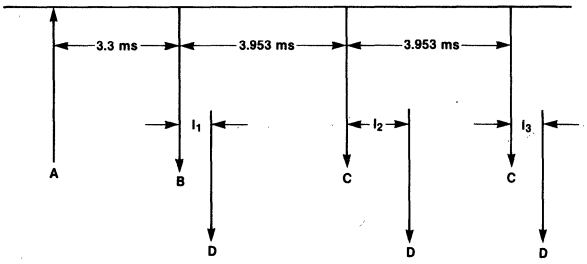
Figure 10 illustrates a possible signal sequence for a timer that is initially loaded with 200, then allowed to run continuously.

F3851 Interrupt Control Register

The interrupt control register (ICR) has the I/O port address XXXXXX10 (where XXXXXX is the 6-bit I/O port address select). Data is loaded into this register (I/O port) using an OUT or OUTS instruction. Data cannot be read out of this register. The contents of the ICR are interpreted as follows:

Contents of I/O Port	Interpretation
B'XXXXXX00'	Disable all interrupts
B'XXXXXX01'	Enable external interrupt, disable timer interrupt
B'XXXXXX10'	Disable all interrupts
B'XXXXXX11'	Disable external interrupt, enable timer interrupt

Figure 10 Time-Out and Interrupt Request Timing



A — 200 LOADED INTO TIMER
B — FIRST TIME OUT
C — SECOND AND SUBSEQUENT TIME-OUTS
D — INTERRUPT SERVICE ROUTINES BEING ENTERED BY CPU
I_1, I_2, I_3 — INTERVALS BETWEEN TIME-OUT INTERRUPT REQUEST REACHING INTERRUPT LOGIC AND SERVICE ROUTINES BEING ENTERED BY CPU

Table 3 F3851 Timer Counts

Contents of Counter	Counts to Interrupt	Contents of Counter	Counts to Interrupt	Contents of Counter	Counts to Interrupt	Contents of Counter	Counts to Interrupt
FE	254	4D	189	D2	124	9F	59
FD	253	9A	188	A5	123	3D	58
FB	252	34	187	4B	122	7C	57
F7	251	69	186	96	121	F8	56
EE	250	D3	185	2D	120	F1	55
DC	249	A7	184	5B	119	E2	54
B8	248	4F	183	B7	118	C5	53
71	247	9E	182	6E	117	8A	52
E3	246	3C	181	DD	116	15	51
C7	245	78	180	BA	115	2A	50

Table 3 F3851 Timer Counts

8E	244	F0	179	75	114	55	49
1D	243	E0	178	EB	113	AA	48
3B	242	C1	177	D6	112	54	47
76	241	82	176	AD	111	A8	46
ED	240	04	175	5A	110	50	45
DA	239	06	174	85	109	A0	44
B4	238	12	173	6A	108	41	43
68	237	24	172	D5	107	83	42
D1	236	48	171	AB	106	06	41
A3	235	90	170	56	105	0D	40
47	234	21	169	AC	104	1A	39
8F	233	42	168	58	103	35	38
1F	232	84	167	B1	102	6B	37
3F	231	°A	166	62	101	D7	36
7E	230	14	165	C4	100	AF	35
FC	229	28	164	88	99	5E	34
F9	228	51	163	11	98	BD	33
F3	227	A2	162	22	97	7B	32
E6	226	45	161	44	96	F6	31
CD	225	8B	160	89	95	EC	30
9B	224	17	159	13	94	D8	29
36	223	2E	158	26	93	B0	28
6D	222	5D	157	4C	92	60	27
DB	221	BB	156	98	91	C0	26
B6	220	77	155	30	90	80	25
6C	219	EF	154	61	89	00	24
D9	218	DE	153	C2	88	01	23
B2	217	BC	152	84	87	03	22
64	216	79	151	03	86	07	21
C8	215	F2	150	10	85	0F	20
91	214	E4	149	20	84	1E	19
23	213	C9	148	40	83	3D	18
46	212	93	147	81	82	7A	17
8D	211	27	146	02	81	F4	16
1B	210	4E	145	05	80	E8	15
37	209	9C	144	0B	79	D0	14
6F	208	38	143	16	78	A1	13
DF	207	70	142	2C	77	43	12
BE	206	E1	141	59	76	87	11
7D	205	C3	140	B3	75	0E	10
FA	204	86	139	66	74	1C	9
F5	203	0C	138	CC	73	39	8
EA	202	18	137	99	72	72	7
D4	201	31	136	32	71	E5	6
A9	200	63	135	65	70	CB	5
52	199	C6	134	CA	69	97	4
A4	198	8C	133	95	68	2F	3
49	197	19	132	2B	67	5F	2
92	196	33	131	57	66	BF	1
25	195	67	130	AE	65	7F	0
4A	194	CE	129	5C	64	FE	254
94	193	9D	128	B9	63		
29	192	3A	127	73	62		
53	191	74	126	E7	61		
A6	190	E9	125	CF	60		

In these I/O port contents definitions, X represents "don't care" binary digits.

F3856 Timer and Interrupt Control Registers

The F3856 logic responds to an interrupt request that can originate internally from the timer logic or from input by an external device, or from the pulse width measurement circuits. Interrupt functions present in the F3856 include the ability to program the active transition of the external interrupt, the ability to have both the timer and the external interrupts active at the same time, and the ability to measure pulse width of an external signal.

The timer is an 8-bit binary count-down register that is used in conjunction with interrupt logic to generate real-time intervals, to measure elapsed time between two events, or to measure a pulse width appearing on the $\overline{EXT}$ $\overline{INT}$ signal. The timer is selected to run in one of four values provided by the prescaler and can be made to start counting or stop counting under program control. Also, the timer contents can be read back under program control.

A zero-detect circuit in the timer detects transitions from a one-count to a zero-count and provides a signal to the interrupt circuits. If all other conditions are satisfied, interrupt circuits, after receiving this signal, request an interrupt service from the CPU.

An external interrupt can be selected under program control to detect the falling or rising edge of the signal. The active edge is determined by the contents in a bit in the interrupt control register.

Both interrupts can be enabled at the same time. When both interrupts are enabled, they are serviced on a first-come, first-served basis. For example, if the timer interrupt arrives later than the unserviced external interrupt, the external interrupt is serviced first, and the timer interrupt remains stored until it is serviced or cleared. If both interrupts arrive at the same cycle, the timer interrupt is handled first.

The internal timer register (TR) and interrupt control register (ICR) are associated with the two high address ports. The TR, depending on various functions, is in one of two modes: stationary or run. In the stationary mode, the contents of the TR remain unaffected. In the run mode, the TR is a binary count-down register, which decrements every 2, 8, 32, or 128 ϕ clock time, depending on the value of the two prescaler bits on the ICR. A circuit detects the one-count-to-zero-count transition of the register and stores it in a flip-flop for interrupt purposes. This flip-flop is cleared any time a new value is loaded into TR.

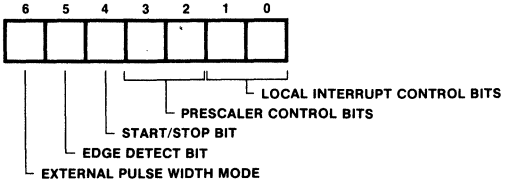
The flip-flop is not cleared by a loading of ICR. While counting, the timer jumps from all-zero value to all-one value and, depending on prescaler values, provides an interrupt period of every 512, 2048, 8192, or 32768 ϕ clocks.

If the timer is in the run mode and the ICR is set for a prescaler value of 2 at the time a value of 2 or 1 is loaded into the TR, the next transition from a one-count to a zero-count is not detected.

F3856 Interrupt Control Register Configuration
The ICR is a 7-bit register used to define various modes of interrupt, the value of the prescaler, and external pulse width measurement. This register is loaded by output instructions; no provision is made to read the contents of this register. The ICR, along with the I/O ports on the F3856, is reset to zero during the reset sequence.

The configuration of this register is shown in figure 11.

Figure 11 F3856 ICR Configuration



Local Interrupt Control (Bits 0-1)—These modes define the interrupt state of the timer and external interrupts (see table 4).

Table 4 F3856 Timer and External Interrupt Modes

Bit 1	Bit 0	Function
0	0	No Interrupt
0	1	Enable External Interrupt Only
1	0	Enable Both External and Timer Interrupts
1	1	Enable Timer Interrupt Only

Prescaler Control (Bits 2-3)—These bits define one of the four different prescalers for the timer (refer to table 5).

Table 5 F3856 Timer Prescaler Modes

Bit 3	Bit 2	Prescaler Value	Timer Resolution at 2 MHz	Timer Period at 2 MHz
1	1	2	1 μs	256 μs
1	0	8	4 μs	1.024 ms
0	0	32	16 μs	4.095 ms
0	1	128	64 μs	16.384 ms

Start-Stop Timer (Bit 4)—This bit controls the TR. When at 0, the TR is in the run mode; when at 1, the TR is in the stationary mode.

Edge Detect Control (Bit 5)—This bit defines the active edge of the $\overline{\text{EXT INT}}$ input signal as the source during external interrupts. When this bit is at 0, the falling edge is active; when it is at 1, the rising edge is active.

External Pulse Width Mode (Bit 6)—When this bit is at 0, no special function is performed and the interrupts and timer circuits are controlled by bits 0 through 5 of the ICR. However, when this bit is at 1, the special function of pulse width measurement is performed.

Pulse Width Measurement

The following procedure is used to measure pulse width for the F3856 PSU (refer to figure 12).

1. Before the pulse arrives, set the ICR as follows:
   a. Set the external pulse width mode bit to 1.
   b. Set the edge detect bit to 1 for a negative pulse or to 0 for a positive pulse.
   c. Set the start/stop bit to 1 (stop mode).
   d. Set the prescaler bits to the value of prescaler desired.
   e. Set the interrupt bits to turn on both interrupts.
2. Load TR with an initial value.
3. As soon as the pulse arrives, the timer starts counting and provides the timer interrupts at zero crossing.
4. At the end of the pulse, the timer stops counting and provides an external interrupt, indicating the end of the pulse. The timer contents can now be read under program control for calculating the pulse width.

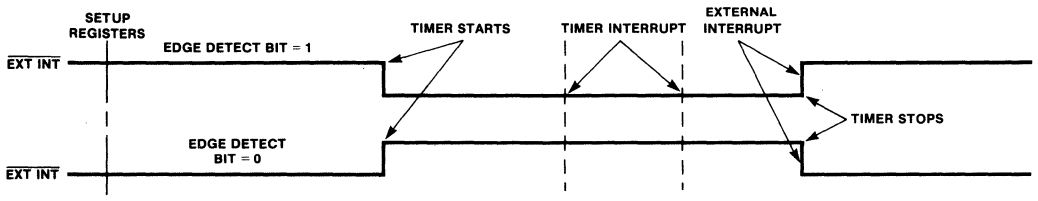
In this procedure, both interrupts are enabled. It is possible to disable one or both interrupts. If the external interrupt is not enabled, the timer stops at the end of the pulse. However, some means of indication are necessary to detect the end of the pulse to the main program. If the timer interrupt is not enabled, the timer zero crossing is not detected. If the pulse duration is always short, such that the timer is stopped before reaching zero, it is not necessary to enable the timer interrupt.

When the timer is loaded with a zero count, the timer interrupt does not occur immediately, although the timer is a zero-count. The timer interrupt occurs only after the one-to-zero transition during the countdown. Hence, when the timer is loaded with a zero count, the timer interrupt occurs after 256 timer counts.

This feature of being able to load a zero count in the timer without getting interrupted allows the programmer to have complete control over the timer count and is also useful during the pulse width measurement mode.

During reset procedures, the ICR is loaded with zero, which disables the local interrupt controls and establishes the trailing edge of the $\overline{\text{EXT REQ}}$ input signal as the active edge for the external interrupt. The active edge of the external signal can be changed by bit 5 of the ICR. However, when this bit is changed, and the level appearing on the external signals is of the same level as the one obtained after the new active edge, an external interrupt is generated. For example, when changing the active edge of the external signal from trailing edge to rising edge under program control, if the external signal is already at a high level, an interrupt is generated.

Figure 12 F3856 Pulse Width Measurement



If such interrupts are undesirable, an additional step is necessary to disable the local external interrupt control during the change of ICR bit 5. For example, when loading the ICR for the change of direction, the external interrupt control can be disabled with the same instruction, and the next instruction can then enable it.

Note that the feature of generating an interrupt by changing bit 5 of the ICR can be used for software (program-generated) interrupts.

PSU Interrupt Handling

A typical F8 system interrupt interconnection is shown in figure 13. Each PSU and PIO has a $\overline{\text{PRI}}$ $\overline{\text{IN}}$ and a $\overline{\text{PRI}}$ $\overline{\text{OUT}}$ line so that they can be daisy-chained together in any order to form a priority level of interrupts. Depending on the contents of the ICR, the interrupt control logic can be accepting timer interrupts or external interrupts, or neither, but never both.

Figure 14 is a diagram of the PSU interrupt logic. Between the $\overline{\text{EXT}}$ $\overline{\text{INT}}$ input signal or the time-out input and the $\overline{\text{INT}}$ $\overline{\text{REQ}}$ output signal, there are three flip-flops. The $\overline{\text{EXT}}$ $\overline{\text{INT}}$ signal and the time-out interrupt input each have a synchronizing flip-flop and edge detect logic.

Each edge detect clock is followed by its own interrupt flip-flop that latches the true condition.

The outputs of the timer interrupt flip-flop and the external interrupt flip-flop are ORed to set the service request flip-flop, providing that an interrupt from some other PSU is not being acknowledged.

The $\overline{\text{INT}}$ $\overline{\text{REQ}}$ signal is the NAND of priority input and service request. This is an open-drain signal. The $\overline{\text{INT}}$ $\overline{\text{REQ}}$ signal of several PSUs can be tied together so that any one can force the line to 0 V if it is requesting interrupt service; a pull-up to V_{DD} is provided by the F3850 CPU to the $\overline{\text{INT}}$ $\overline{\text{REQ}}$ input pin.

The $\overline{\text{PRI}}$ $\overline{\text{IN}}$ signal is part of the interrupt priority chain. The chain begins by a strap to V_{SS}. Each device in the chain has a $\overline{\text{PRI}}$ $\overline{\text{IN}}$ input signal and a $\overline{\text{PRI}}$ $\overline{\text{OUT}}$ output signal. The $\overline{\text{PRI}}$ $\overline{\text{OUT}}$ signal of the PSU is active (0 V) only if the $\overline{\text{PRI}}$ $\overline{\text{IN}}$ signal is active (0 V) and service request is inactive This means that the $\overline{\text{PRI}}$ $\overline{\text{OUT}}$ and $\overline{\text{INT}}$ $\overline{\text{REQ}}$ signals are always at opposite levels. The $\overline{\text{PRI}}$ $\overline{\text{OUT}}$ signal becomes the $\overline{\text{PRI}}$ $\overline{\text{IN}}$ signal for the next device in the interrupt priority daisy chain, if there is one. The function of the priority daisy chain is to ensure that just one device at a time is requesting interrupt service.

The service request flip-flop cannot become set if another interrupt request is being acknowledged anywhere in the system. Rather, if an interrupt request has been latched into the timer interrupt flip-flop or the external interrupt flip-flop, the PSU logic waits until after the process of acknowledging the other interrupt has been completed before setting the service request. This precaution is necessary to ensure that the priority chain is not altered during acknowledgement; an error would occur if one half of the interrupt vector came from one device and the second half from some other device.

The service request flip-flop is cleared after an interrupt from the PSU has been acknowledged. It is also cleared whenever the interrupt control register for the PSU is accessed by an output instruction.

The conditions for setting the timer interrupt flip-flop and the external interrupt flip-flop differ slightly. External interrupts must be enabled before the external interrupt flip-flop can be set by a negative-going transition of the $\overline{\text{EXT}}$ $\overline{\text{INT}}$ signal. However, the timer interrupt flip-flop is set by a timer time-out independent of the timer interrupt enable bit. This means that the PSU can detect a time-out interrupt that is requested while the PSU was checking for external interrupts.

The timer interrupt flip-flop is cleared whenever the PSU device timer is loaded or when its timer interrupt has been acknowledged. The external interrupt flip-flop is cleared whenever the device interrupt control register is accessed by an output instruction or when its external interrupt has been acknowledged.

Interrupt Acknowledge Sequence

Upon receiving an interrupt request, whether from an external source through the $\overline{\text{EXT}}$ $\overline{\text{INT}}$ signal or from the internal timer, the PSU and CPU go through an interrupt sequence that ultimately results in the execution of an interrupt service routine located at the memory address indicated by the interrupt address vector. Figures 15 and 16 illustrate the interrupt sequences for the two cases. Events occurring in these sequences are labeled A through H.

Event A—The initial interrupt request arrives. The falling edge of the $\overline{\text{EXT}}$ $\overline{\text{INT}}$ pin identifies an external interrupt. The rising edge of the interval timer output indicates a time-out.

Event B—The synchronizing flip-flop in the PSU control logic changes state.

Event C—The timer or external interrupt flip-flop goes true, indicating the local interrupt logic acknowledgement of the interrupt. The timer interrupt flip-flop always responds and saves the time-out occurrence, whereas the external inter-

Figure 13 F8 System Interrupt Interconnection

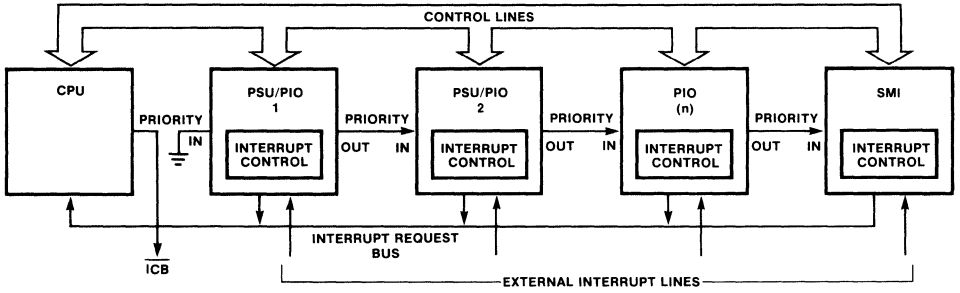
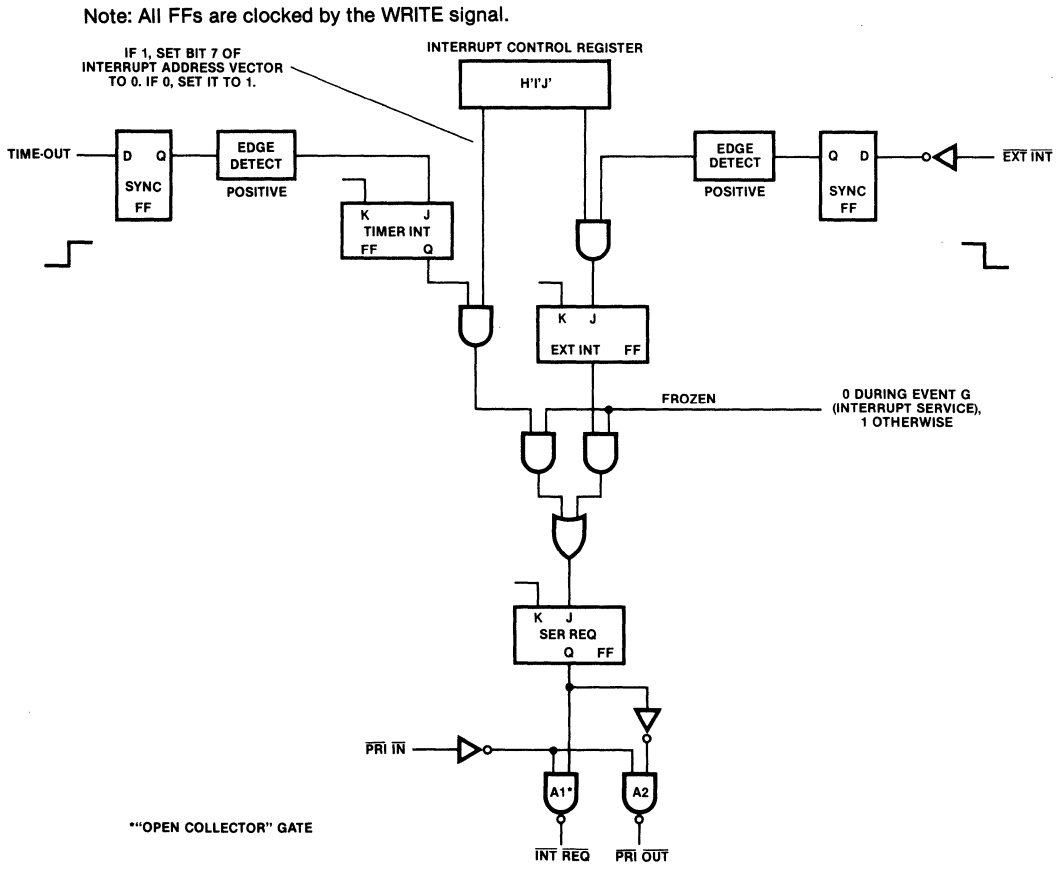


Figure 14 Conceptual Illustration of F3851 PSU Interrupt Logic



rupt flip-flop is set at this time only if the external interrupt mode is enabled within the local control logic.

Event D—The INT REQ line is pulled low by the PSU, passing the request for servicing on to the CPU. The following conditions must be present for this to occur:

1. The PRI IN pin must be low.
2. The proper enable state must exist in the local control logic for the type of interrupt (timer or external).
3. The system is not already into Event F because of servicing some other interrupt.

Event E—The CPU now begins its response to the $\overline{\text{INT REQ}}$ line by transmitting the unique ROMC state H'10'.

This occurs only when the following conditions are satisfied:

1. The CPU is executing the last cycle of an instruction (beginning an instruction fetch).
2. The ICB is enabled (ICB = 0).
3. The current instruction fetch is not protected.

Event F—The CPU generates the interrupt acknowledge sequence of ROMC states.

Event G—At this point, the CPU begins fetching the first instruction of the interrupt service routine. In the PSU interrupt logic, the service request flip-flop and the appropriate interrupt request flip-flop have been cleared.

Event H—The CPU begins executing the first instruction of the interrupt service routine.

Interrupt Address Vector

During the interrupt acknowledge, the interrupting PSU provides a 16-bit interrupt address vector (refer to figure 17). The CPU causes this vector to be loaded into PC_0 so that program execution can branch to the routine that handles this particular interrupt. Fifteen bits of the interrupt vector are specified as a mask option. Bit 7 cannot be masked; it is set by the interrupt control logic to 0 if the timer interrupt is enabled or to 1 if the external interrupt is enabled.

Figure 15 Timer Interrupt Sequence

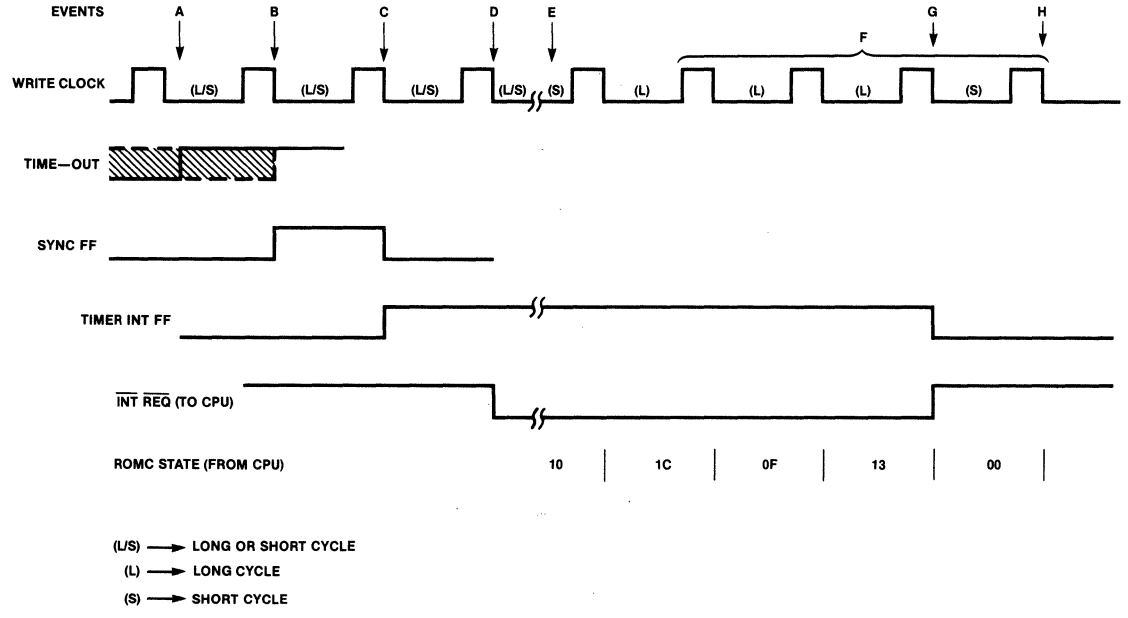


Figure 16 External Interrupt Sequence

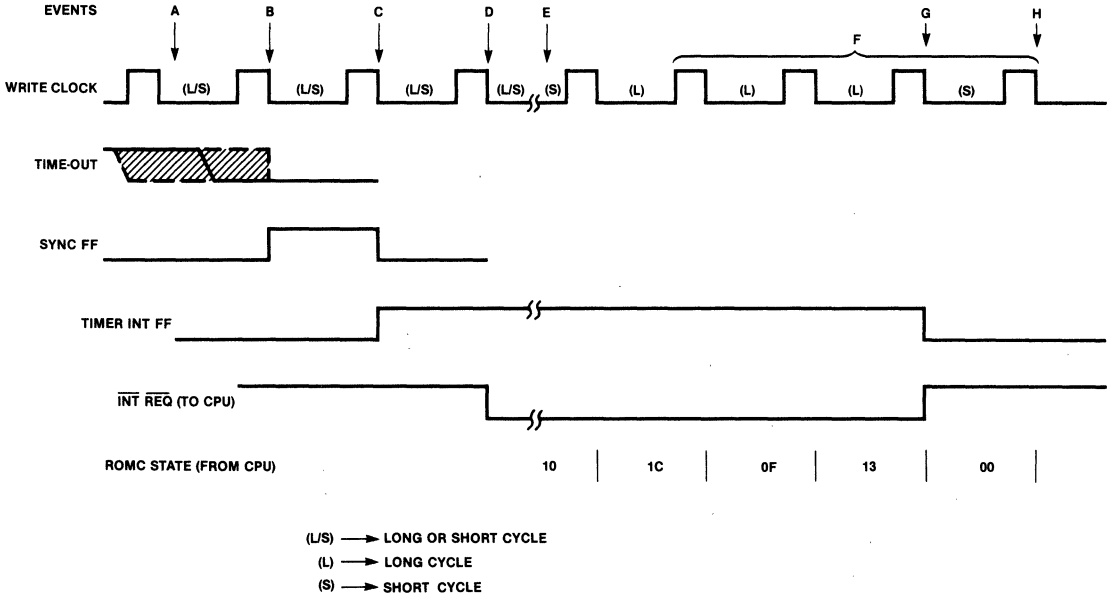
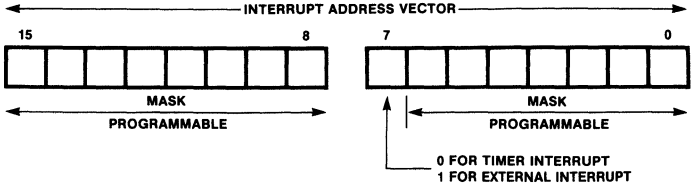


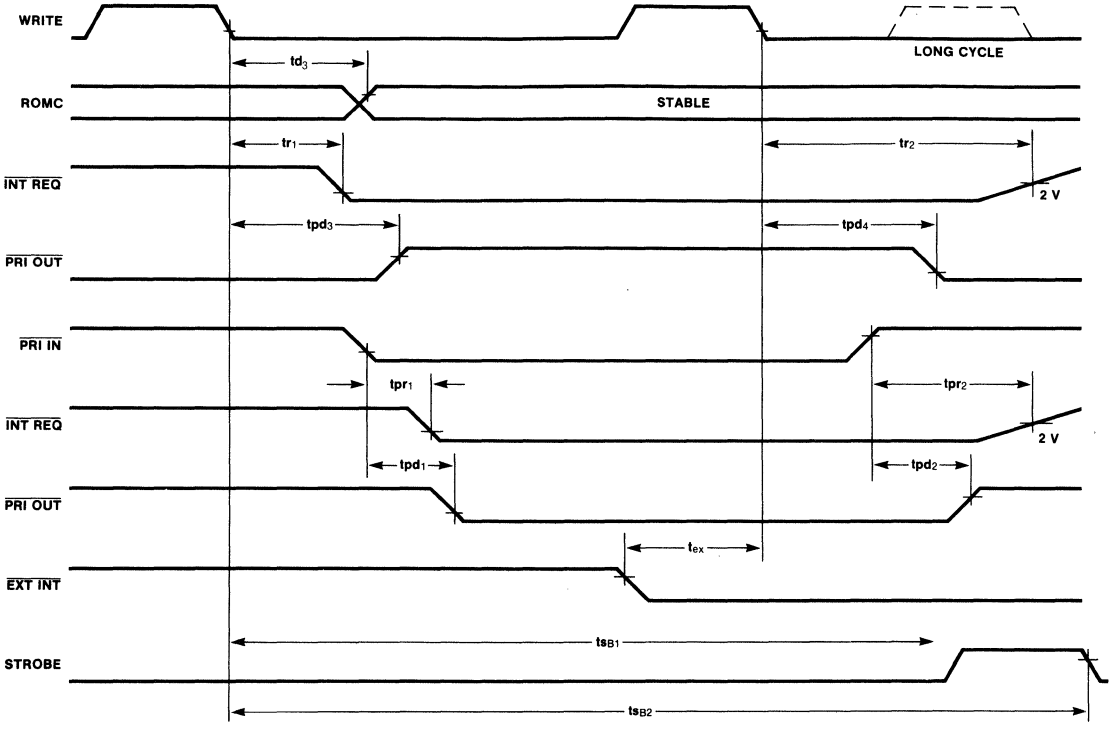
Figure 17 Interrupt Address Vector



Interrupt Signal Timing
Timing for signals associated with the PSU interrupt logic is shown in figure 18. All signal characteristics are given in the timing characteristics section of this data sheet.

Note: Timing measurements are made at valid logic level to valid logic level of the signals referenced unless otherwise noted.

Figure 18 PSU Interrupt Timing



Timing Characteristics

The timing characteristics of the PSU devices are described in table 6. The ac characteristics are $V_{SS} = 0$ V, $V_{DD} = +5.0$ V ±5%, $V_{GG} = +12$ V ±5%, $T_A = 0°C$ to $+70°C$, unless otherwise specified.

Table 6 PSU Signal Timing Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Test Conditions
$P\phi$	ϕ Period	0.5		10	μs	
PW_1	ϕ Pulse Width	180		$P\phi-180$	ns	t_r, $t_f = 50$ ns typ.
td_1, td_2	ϕ to Write + Delay			250	ns	$C_L = 100$ pF
td_4	WRITE to DB Input Delay			$2P\phi+1.0$	μs	
PW_2	WRITE Pulse Width	$P\phi-100$		$P\phi$	ns	t_r, $t_f = 50$ ns typ.
PW_S	WRITE Period; Short		$4P\phi$			
PW_L	WRITE Period; Long				ns	
td_3	WRITE to ROMC Delay			550	ns	
td_7	WRITE to DB Output Delay WRITE to $\overline{DBDR}$ – Delay	$2P\phi+100-td_2$	$2P\phi+200$	$2P\phi+850-td_2$	ns	$C_l = 100$ pF
td_8	WRITE to $\overline{DBDR}$ + Delay		200		ns	Open drain
tr_1	WRITE to $\overline{INT}$ $\overline{REQ}$ – Delay			430	ns	$C_L = 100$ pF(1)
tr_2	WRITE to $\overline{INT}$ $\overline{REQ}$ + Delay			430	ns	$C_l = 100$ pF(3)
tpr_1	$\overline{PRI}$ $\overline{IN}$ to INT $\overline{REQ}$ – Delay		200		ns	$C_L = 100$ pF(2)
tpd_1, tpd_2	$\overline{PRI}$ $\overline{IN}$ to $\overline{PRI}$ $\overline{OUT}$ Delay		800		ns	$C_L = 50$ pF
tpd_3, tpd_4	WRITE to $\overline{PRI}$ $\overline{OUT}$ Delay		600		ns	$C_L = 50$ pF
t_{sp}	WRITE to Output Stable			1.0	μs	$C_L = 50$ pF, standard pull-up(3)
t_{od}	WRITE to Output Stable			2.5	μs	$C_L = 50$ pF, $R_L = 12.5$ kΩ, open drain(5)
t_{dp}	WRITE to Output Stable		200	400	ns	$C_L = 50$ pF, driver pull-up
t_{su}	I/O Set-up Time	1.3			μs	
t_h	I/O Hold Time	0			ns	
t_{ex}	$\overline{EXT}$ $\overline{INT}$ Set-up Time	400			ns	
t_sB_1	WRITE to STROBE + Delay			$5P\phi+300$	ns	$C_L = 50$ pF
t_sB_2	WRITE to STROBE – Delay			$6P\phi+410$	ns	$C_L = 50$ pF

Notes
1. Assume priority in was enabled ($\overline{PRI}$ $\overline{IN} = 0$) in the previous F8 cycle, before the interrupt is detected in the PSU.
2. The PSU has an interrupt pending before priority in is enabled.
3. Assume pin tied to $\overline{INT}$ $\overline{REQ}$ input of the F3850 CPU.
4. Input and output capacitance is 3 to 5 pF, typical, on all pins except V_{DD}, V_{GG}, and V_{SS}.

DC Characteristics

The dc characteristics of the PSU devices are provided in tables 7 and 8.

Supply Currents $V_{SS} = 0$ V, $V_{DD} = +5$ V ±5%, $V_{GG} = +12$ V ±5%, $T_A = 0°C$, $+70°C$

Symbol	Parameter	Min.	Typ.	Max.	Units	Test Conditions
I_{DD}	V_{DD} Current		28	60	mA	f = 2 MHz, outputs unloaded
I_{GG}	V_{GG} Current		10	30	mA	f = 2 MHz, outputs unloaded

Table 7 F3851 PSU DC Characteristics

Symbol	Parameter	Signal	Min.	Max.	Units	Test Conditions
V_{IH}	Input High Voltage	Data Bus (DB_0 -DB_7)	2.9	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
V_{OH}	Output High Voltage		3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 1.6$ mA
I_{IH}	Input High Current			1.0	μA	$V_{IN} = V_{DD}$, 3-state mode
I_{OL}	Input Low Current			−1.0	μA	$V_{IN} = V_{SS}$, 3-state mode
V_{IH}	Input High Voltage	Clock Lines (ϕ, WRITE)	4.0	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_L	Leakage Current			3.0	μA	$V_{IN} = V_{DD}$
V_{IH}	Input High Voltage	Priority In and Control	3.5	V_{DD}	V	
V_{IL}	Input Low Voltage	Lines ($\overline{PRI}$ $\overline{IN}$, $ROMC_0$ -	V_{SS}	0.8	V	
I_L	Leakage Current	$ROMC_4$)		3.0	μA	$V_{IN} = V_{DD}$
V_{OH}	Output High Voltage	Priority out ($\overline{PRI}$ $\overline{OUT}$)	3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 100\ \mu A$
V_{OH}	Output High Voltage	Interrupt Request			V	Open-drain output[1]
V_{OL}	Output Low Voltage	($\overline{INT}$ $\overline{REQ}$)	V_{SS}	0.4	V	$I_{OL} = 1$ mA
I_L	Leakage Current			3.0	μA	$V_{IN} = V_{DD}$
V_{OH}	Output High Voltage	Data Bus Drive ($\overline{DBDR}$)				External pull-up
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 2$ mA
I_L	Leakage Current			3.0	μA	$V_{IN} = V_{DD}$
V_{IH}	Input High Voltage	External Interrupt	3.5		V	
V_{IL}	Input Low Voltage	($\overline{EXT}$ $\overline{INT}$)		0.8	V	
V_{IC}	Input Clamp Voltage			15	V	$I_{IH} = 185\ \mu A$
I_{IH}	Input High Current			10	μA	$V_{IN} = V_{DD}$
I_{IL}	Input Low Current			−225	μA	$V_{IN} = 2$ V
I_{IL}	Input Low Current		−150	−500	μA	$V_{IN} = V_{SS}$
V_{OH}	Output High Voltage	I/O Port Option A	3.9[5]	V_{DD}	V	$I_{OH} = -30\ \mu A$
V_{OH}	Output High Voltage	(Standard Pull-up)	2.9	V_{DD}	V	$I_{OH} = -150\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 1.6$ mA
V_{IH}	Input High Voltage		2.9[3]	V_{DD}	V	Internal pull-up to V_{DD}[3]
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_L	Leakage Current			1.0	μA	$V_{IN} = V_{DD}$
I_{IL}	Input Low Current			−1.6	mA	$V_{IN} = 0.4$ V[4]
V_{OH}	Output High Voltage	I/O Port Option B				External pull-up
V_{OL}	Output Low Voltage	(Open Drain)	V_{SS}	0.4	V	$I_{OL} = 2$ mA
V_{IH}	Input High Voltage		2.9[3]	V_{DD}	V	[3]
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_{IL}	Leakage Current			2.0	μA	$V_{IN} = +12$ V
V_{OH}	Output High Voltage	I/O Port Option C	3.75	V_{DD}	V	$I_{OH} = -1$ mA
V_{OL}	Output Low Voltage	(Driver Pull-up)	V_{SS}	0.4	V	$I_{OL} = 1.6$ mA

Notes

1. Pull-up resistor to V_{DD} on CPU.
2. Positive current is defined as conventional current flowing into the pin referenced.
3. Hysteresis input circuit provides additional 0.3 V noise immunity while internal/external pull-up provides TTL compatibility.
4. Measured while I/O port is outputting a high level.
5. Guaranteed but not tested.

Table 8 F3856 PSU DC Characteristics

Symbol	Parameter	Signal	Min.	Max.	Units	Test Conditions
V_{IH}	Input High Voltage	Data Bus (DB_0 -DB_7)	2.9	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
V_{OH}	Output High Voltage		3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 1.6$ mA
I_{IH}	Input High Current			3.0	μA	$V_{IN} = V_{DD}$, 3-state mode
I_{OL}	Input Low Current			−3.0	μA	$V_{IN} = V_{SS}$, 3-state mode
V_{IH}	Input High Voltage	Clock Lines (ϕ, WRITE)	4.0	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_L	Leakage Current			3.0	μA	$V_{IN} = V_{DD}$
V_{IH}	Input High Voltage	Priority In and Control	3.5	V_{DD}	V	
V_{IL}	Input Low Voltage	Lines ($\overline{PRI}$ $\overline{IN}$, $ROMC_0$ -	V_{SS}	0.8	V	
I_L	Leakage Current	$ROMC_4$)		3.0	μA	$V_{IN} = V_{DD}$
V_{OH}	Output High Voltage	Priority out ($\overline{PRI}$ $\overline{OUT}$)	3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 100\ \mu A$
V_{OH}	Output High Voltage	Interrupt Request			V	Open-drain output[1]
V_{OL}	Output Low Voltage	($\overline{INT}$ $\overline{REQ}$)	V_{SS}	0.4	V	$I_{OL} = 1.0$ mA
I_L	Leakage Current			3.0	μA	$V_{IN} = V_{DD}$
V_{OH}	Output High Voltage	Data Bus Drive ($\overline{DBDR}$)				External pull-up
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 2.0$ mA
I_L	Leakage Current			3.0	μA	$V_{IN} = V_{DD}$
V_{OH}	Input High Voltage	Strobe	3.9	V_{DD}	V	I_{OH} = -1.0 mA
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	I_{OL} = 2.0 mA
V_{IH}	Input High Voltage	External Interrupt	2.9	V_{DD}	V	$I_{IN} = -130\ \mu A$ (internal pull-up)
V_{IL}	Input Low Voltage	($\overline{EXT}$ $\overline{INT}$)	V_{SS}	0.8	V	
I_{IL}	Input Low Current			−1.6	mA	$V_{IN} = 0.4$ V
V_{OH}	Output High Voltage	I/O Port Option A	3.9	V_{DD}	V	$I_{OH} = -30\ \mu A$[5]
V_{OH}	Output High Voltage	(Standard Pull-up)	2.9	V_{DD}	V	$I_{OH} = -150\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 1.6$ mA
V_{IH}	Input High Voltage		2.9	V_{DD}	V	Internal pull-up to V_{DD}[3]
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_{IL}	Input Low Current			−1.6	mA	$V_{IN} = 0.4$ V[4]
V_{OH}	Output High Voltage	I/O Port Option B				External pull-up
V_{OL}	Output Low Voltage	(Open Drain)	V_{SS}	0.4	V	$I_{OL} = 2$ mA[3]
V_{IH}	Input High Voltage		2.9	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
V_{OH}	Output High Voltage	I/O Port Option C	4.0	V_{DD}	V	$I_{OH} = -1.0$ mA
V_{OL}	Output Low Voltage	(Driver Pull-up)	V_{SS}	0.4	V	$I_{OL} = 2.0$ mA

Notes
1. Pull-up resistor to V_{DD} on CPU.
2. Positive current is defined as conventional current flowing into the pin referenced.
3. Hysteresis input circuit provides additional 0.3 V noise immunity while internal/external pull-up provides TTL compatibility.
4. Measured while I/O port is outputting a high level.
5. Guaranteed but not tested.

Mask Options

The ROM array may contain object program code, tables of nonvarying data, or both. Every PSU is implemented using a custom mask that specifies the state of every ROM bit, as well as certain address mask options that are external to the ROM array. The following mask options are specified:

1. The 1024 or 2048 bytes of ROM storage. This reflects programs and permanent data table stored in the PSU memory.
2. The 5-bit or 6-bit page select. This defines the PSU address space.
3. The 6-bit I/O port address select. This defines the four PSU I/O port addresses.
4. The 16-bit interrupt address vector, excluding bit 7.
5. The I/O port output option. The choices are the standard pull-up (option A), the open-drain (option B), and the driver pull-up (option C).

PSU Mask Option Formats

The format for mask options must be submitted to Fairchild Microprocessor Division before device manufacture. The data to be stored in permanent memory may be submitted in the form of an EPROM or HP2644/HP2645 cartridge (Formulator format only). Other options must be specified on the Fairchild ROM Code Entry Form, available from a Fairchild representative.

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Supply Voltage V_{GG}	−0.3 V, +15 V
Supply Voltage V_{DD}	−0.3 V, +7 V
I/O Port Open Drain Option	−0.3 V, +15 V
External Interrupt Input (F3851)	−600 μA, +225 μA
Other I/O Port Options	−0.3 V, +7 V
All Inputs and Outputs	−0.3 V, +7 V
Storage Temperature	−55°C, +150°C
Operating Temperature	0°C, +70°C

Thermal Resistance Values

Plastic

θ_{JA} (Junction to Ambient) = 60°C/W (Still Air)
θ_{JC} (Junction to Case) = 42°C/W

Ceramic

θ_{JA} (Junction to Ambient) = 48°C/W (Still Air)
θ_{JC} (Junction to Case) = 33°C/W

Recommended Operating Ranges

The recommended operating ranges of the PIO devices are shown below.

Symbol	Parameter	Min.	Typ.	Max.
V_{DD}	Supply Voltage	+4.75 V	+5 V	+5.25 V
V_{GG}	Supply Voltage	+11.4 V	+12 V	+12.6 V
V_{SS}	Ground		0 V	

Ordering Information

Part Number	Package	Temperature Range*
F3851DC	Ceramic	C
F3851DM	Ceramic	M
F3851PC	Plastic	C
F3856DC	Ceramic	C
F3856DM	Ceramic	M
F3856PC	Plastic	C

*C = Commercial Temperature Range 0° to +70°C
L = Limited Temperature Range −40°C to +85°C
M = Military Temperature Range −55°C to +125°C

FAIRCHILD
A Schlumberger Company

F3861
Peripheral Input/Output

Microprocessor Product

Description

The Fairchild F3861 Peripheral Input/Output (PIO) device provides two 8-bit I/O ports, external interrupt, and a programmable timer. An 8-bit wide bidirectional data bus transfers I/O data bytes between the F3870 Central Processing Unit (CPU) and the PIO.

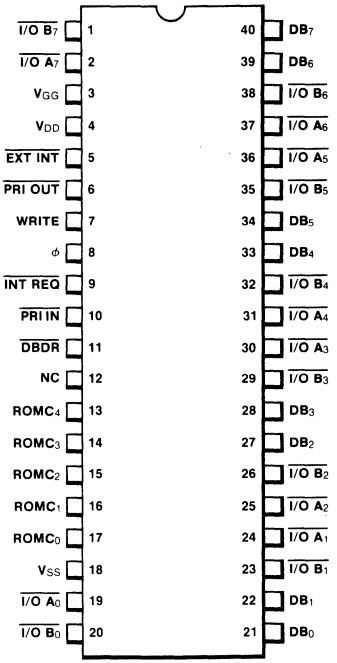
The PIO is used in systems that require the I/O capability and interrupt functions of the F3851 PSU but do not need the read-only memory (ROM) storage of the PSU. The PIO is pin-compatible with the PSU.

The F3861 PIO has five versions available, each with its own set of preassigned I/O port addresses and interrupt vectors.

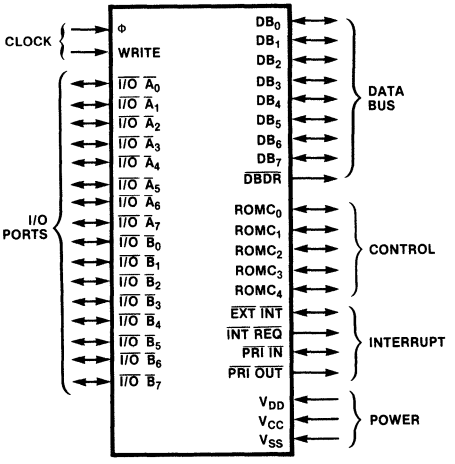
The F3861 is manufactured using isoplanar N-channel, silicon-gate technology; therefore, power dissipation is very low (less than 250 mW).

- **16 Bidirectional, Individualy Controlled I/O Lines Organized as Two 8-bit Ports**
- **Programmable Timer--Preset, Start, Stop, and Read-Back Ability; Selectable Timer Count Rates**
- **Full Interrupt Level--Daisy-Chain Expandable, Independent Interrupt Address Vectors for Timer and External Interrupt**
- **Pulse Width Measurement Capability**
- **TTL and LSTTL Compatible**
- **+5 V and +12 V Power Supplies**
- **2-MHz Operation**
- **Low Power Dissipation, Typically Less Than 250 mW**

Connection Diagram



Signal Functions



Device Organization

The peripheral input/output device includes I/O logic, timer logic, interrupt logic, data bus logic and control logic, as illustrated in figure 1.

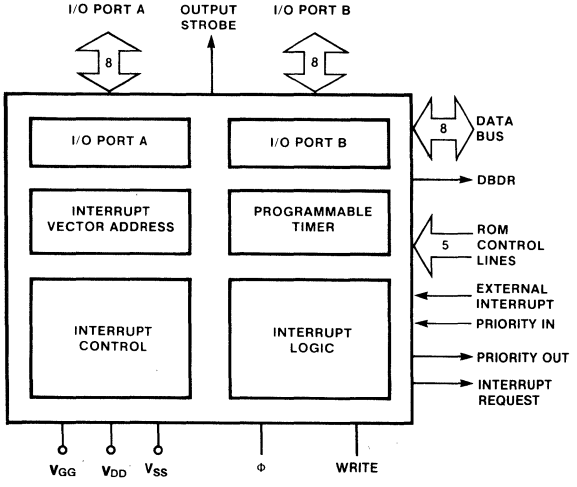
The interrupt logic responds to an interrupt request signal originating from internal timer logic or an external device. Based on priority considerations, the interrupt request is passed on to the F3850 CPU. The programmable timer uses a polynomial shift register in conjunction with interrupt logic to generate real-time intervals.

The 8-bit data bus in the PIO is the main path for transfer of information between the F3850 CPU and other devices in the F8 microprocessor system. The device contains four preassigned I/O port addresses: the two lowest are assigned to the two I/O ports (A and B) and are used to tranfer data to and from external devices. The other two I/O addresses are assigned to two internal registers of the PIO that control interrupt logic and are treated as I/O ports.

Signal Descriptions

The F3861 input and output signals are described in table 1.

Figure 1 F3861 Block Diagram



System Clock Timing

All timing within the F3861 PIO is controlled by the Φ and WRITE signals, which are input from the F3850 CPU. Refer to the F3850 data sheet for a description of these clock signals. The WRITE clock refreshes and updates PIO registers, which are dynamic. The Φ clock also drives the programmable timer.

I/O Ports

The PIO has two bidirectional 8-bit I/O ports used to transmit data between itself and external devices. In binary notation, the address for port A is XXXXXX00 and for port B is XXXXXX01, where the X binary digits are the unique I/O port select code for the PIO (see table 2). For example, if the port select code is 000001, port A may be called port 4 and port B may be called port 5. (The PIO port select code is never designated as all 0s, since ports 0 and 1 are reserved for the F3850 CPU.) In addition, the interrupt control Port (ICP) is addressed as port XXXXXX10 and the binary timer is addressed as port XXXXXX11, which become ports 6 and 7, respectively, for the port select code example given above.

Table 2 F3861 Port Addresses

ADDRESS	ASSIGNED TO
XXXX XX00	I/O Port A
XXXX XX01	I/O Port B
XXXX XX10	Interrupt Control Register
XXXX XX11	Programmable Timer

The port and interrupt address vector assignments are given in table 3.

Table 3 F3861 Port and Address Assignments (HEX)

Version	Port Addresses	Interrupt Address Vector	
		Timer	External
F3861A	4-7	0600	068C
F3861B	8-B	0340	03C0
F3861C	20-23	0320	03A0
F3861D	24-27	0360	03E0
F3861E	4-7	0020	00A0

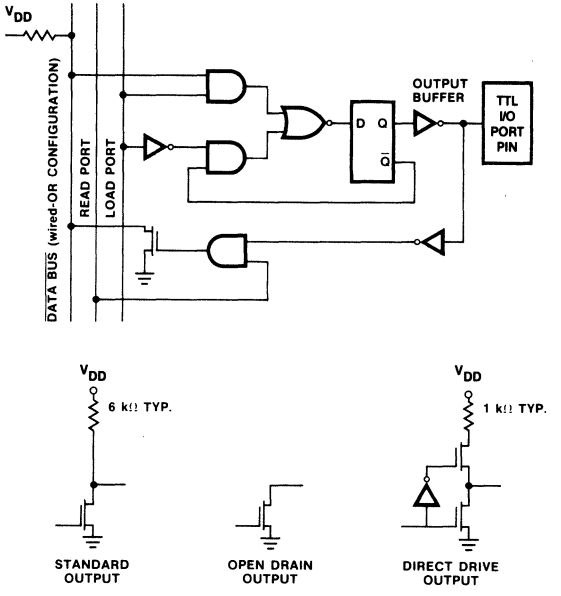
Table 1 F3861 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
Φ	8	Clock	The two clock input signals originate at the F3850 CPU.
WRITE	7		
I/O Ports			
$\overline{I/O\ A_0}$ – $\overline{I/O\ A_7}$	19, 24, 25, 30 31, 36 37, 2	I/O Ports A	Bidirectional ports through which the PIO communicates with logic external to the microprocessor system.
$\overline{I/O\ B_0}$ – $\overline{I/O\ B_7}$	20, 23, 226, 29, 31, 35, 38, 1	I/O Ports B	
Control			
$ROMC_0$ – $ROMC_4$	17, 16, 15, 14, 13	Read Only Memory Control	Input signals that originate at the F3850 CPU and control internal functions of the PIO.
Data Bus			
DB_0 – DB_7	21, 22, 27, 28, 33, 34, 39, 40	Data Bus	Bidirectional three-state lines that link the PIO to all other devices within the microprocessor system.
DBDR	11	Data Bus Drive	A low output, open drain signal that indicates the data bus currently contains data flowing from the PIO.
Interrupt			
$\overline{EXT}$ $\overline{INT}$	5	External Interrupt	A high-to-low transition on this input signal is interpreted as an interrupt request from an external device.
$\overline{INT}$ $\overline{REQ}$	9	Interrupt Request	This output signal is the INT REQ input to the F3850 CPU; it must be output low to interrupt the CPU, which occurs only if PRI IN is low and PIO interrupt control logic is requesting an interrupt.
$\overline{PRI}$ $\overline{IN}$	10	Priority In	Unless this input signal is low, the PIO does not set the INT REQ signal low in response to an interrupt.
$\overline{PRI}$ $\overline{OUT}$	6	Priority Out	This output signal becomes the PRI IN signal to the next device in the interrupt-priority daisy chain; it is output high unless the PRI IN signal is entering the PIO low and the PIO is not requesting an interrupt.
Power			
V_{DD}	4	Power Supply	5 V (± 5%)
V_{GG}	3	Power Supply	+12 V (± 5%)
V_{SS}	18	Ground	System ground—O V; V_{DD} and V_{GG} are referenced to V_{SS}.

Port Pin Description

An output instruction (OUT or OUTS) causes the contents of the CPU accumulator (ACC) to be latched into the addressed port. An input instruction (IN or INS) transfers the contents of the port to the ACC (port 6 is an exception that is described later). The I/O pins on the PIO are logically inverted; the schematic of an I/O pin and available output drive options are shown in figure 2. Each output pin has an output latch that holds the data last output to that pin. The I/O ports of the PIO are configured in the standard pull-up option.

Figure 2 I/O Pin Diagram with Output Buffer Options



Each I/O port pin is a wire-AND structure between an internal output data latch and the external signal. The latch is loaded from the data bus. The output latches are not initialized by the system reset sequence.

When outputting data through an I/O port, the pin can be connected directly to a TTL gate input; data is input to the pin from a TTL device output. Since the I/O pin and the TTL device output are wire-ANDed, it is possible for the state of one to affect the transfer of data out from the I/O pin or in from the TTL device output. In most cases, therefore, I/O port bits should be set for a high level (logic 0) before data input to prevent incoming logic zeros from being masked by logic ones present at the port from previous outputs. However, the ability to mask bits of a port to logic 1 is useful during some input functions.

Programmable Timer

The 8-bit shift register, addressable as an I/O port, functions as a polynomial timer. This timer is loaded with a value of delay; it counts down this value of delay and, after the programmed interval, generates an interrupt through the interrupt logic of the PIO.

The OUT or OUTS instruction is used to load the interval value into the programmable timer; the port number is H'07', H'0B', H'23', or H'27', as appropriate. The timer times out after a time interval given by the product

(period of Φ clock) × (timer counts) × 31

The timer continues to run after a time-out; subsequent time-outs occur at intervals of 7905 Φ clock periods. The timer does not run if it is loaded with the value H'FF'.

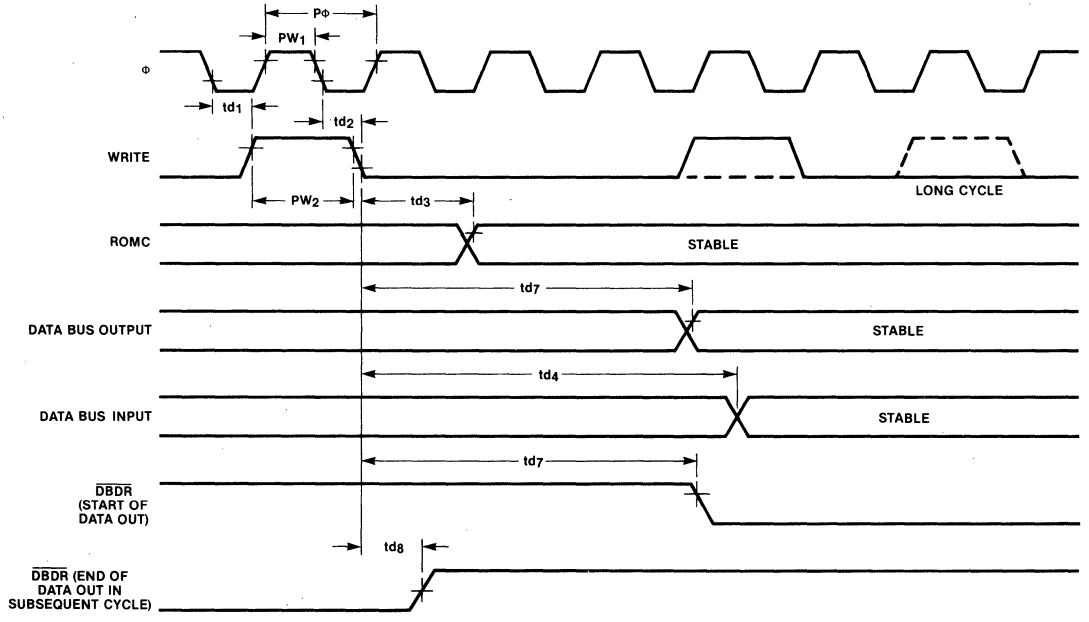
Interrupt Logic

The interrupt logic block is programmed by output instructions to the interrupt control port (port H'06', H'0A', H'22', or H'26', as appropriate). Only the least significant two bits are used; their interpretation is as follows.

Contents of ICP	Interpretation
B'XXXXXX00'	Disable all interrupts
B'XXXXXX01'	Enable external interrupt, disable timer interrupt
B'XXXXXX10'	Disable all interrupts
B'XXXXXX11'	Disable external interrupt, enable timer interrupt

Note: The X designation represents "don't care" binary digits.

Figure 3 PIO Data Bus Timing



Instruction Execution

The PIO responds to signals that are output by the F3850 CPU in the course of implementing instruction cycles. Figure 3 illustrates timing during PIO data output to the data bus. This timing applies whenever a PIO is the data source. The PIO places data on the data bus, even in the worst case, in time for the setup required by any F3850 CPU destination. The PIO receives a byte input from the data bus when commanded by an output instruction to load one of its two I/O ports or internal registers. Data bus timing requirements for input to the PIO are also shown in figure 3; signal characteristics are given in the "Timing Characteristics" section. The data bus drive (DBDR) signal is low while data output by the PIO is stable on the data bus. Thus, a $\overline{\text{DBDR}}$ low signal indicates that the data bus currently contains data flowing from a PIO. For systems with more than one program storage unit (PSU) or PIO, the DBDR output signals may be wire-ORed and the result used as a bus data flow direction indicator. The $\overline{\text{DBDR}}$ signal may remain low until timing interval td of the next instruction cycle following the one in which $\overline{\text{DBDR}}$ was set low.

The PIO device executes the OUT instruction in the same manner as the OUTS instruction; the same is true for the IN and INS instructions. The difference between the long- and short-form instructions is only in the source of the I/O address.

The F8 input/output instructions place the I/O port address on the data bus during one instruction cycle and then use the data bus in the following instruction cycle to do the actual I/O data movement. The read only memory control (ROMC) lines coming from the F3850 CPU signal the PIO that an I/O data movement is occurring during the current instruction cycle. Therefore, the PIO needs to recognize whether the contents of the data bus during the instruction cycle just prior matched any of its four assigned I/O addresses, wherever the ROMC lines indicate an I/O transfer. The address select logic constantly monitors the data bus for a match to any of the four addresses and holds the information of a match through the following cycle.

Input instructions that select a port cause the contents of the selected port to be placed on the data bus during the input cycle. Only the two I/O ports (lowest two addresses)

respond to input instructions. Output instructions that select a port transfer the contents of the data bus to that port. Outputs of the latches change at the end of the I/O transfer cycle.

Interrupt Handling

A typical F8 system interrupt interconnection is shown in figure 4. Each PSU and PIO has a $\overline{PRI}$ $\overline{IN}$ and $\overline{PRI}$ $\overline{OUT}$ line so that they can be daisy-chained together in any order to form a priority level of interrupts. When a PIO receives an interrupt (either timer or external), it pulls its $\overline{PRI}$ $\overline{OUT}$ output signal high, signaling all lower priority peipherals that it has a higher priority interrupt request pending on the CPU. Also, when the PIO device $\overline{PRI}$ $\overline{IN}$ input signal is pulled high by a higher priority peripheral, signaling the PIO that there is a still higher priority interrupt request, it passes that signal along by pulling its $\overline{PRI}$ $\overline{OUT}$ signal high. When the CPU processes an interrupt request, it commands the interrupting device to place its interrupt vector address on the data bus. Only the device with a $\overline{PRI}$ $\overline{IN}$ signal low and an interrupt request pending responds. Should there be another lower priority device with a pending request, it does not respond at that time because its $\overline{PRI}$ $\overline{IN}$ input signal is high.

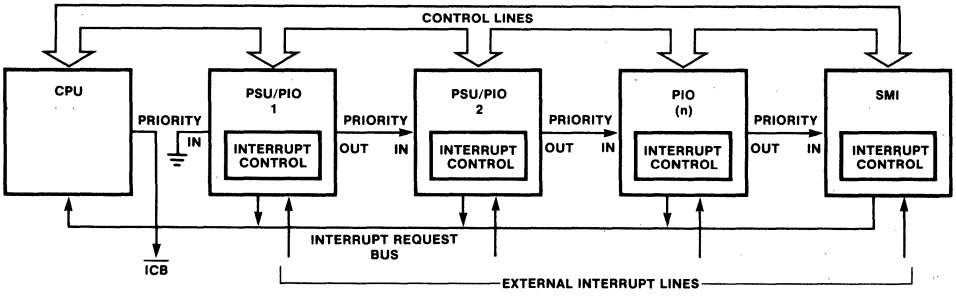
If there are both a timer interrupt request and an external interrupt request when the CPU starts to process the requests, the timer interrupt is handled first.

Within each local interrupt control circuit is a 16-bit interrupt address vector. This vector is the address to which the program counter is set after an interrupt is acknowledged and is therefore the address of the first executable instruction of the interrupt routine.

The interrupt address is unique to the version of the PIO device selected by the user. Fifteen bits are fixed: bits 0 through 6 and bits 8 through 15. Bit 7 (2^7) is dependent on the type of interrupt. This bit is a 0 for internal timer-generated interrupts and a 1 for external interrupts. When the interrupt logic sends an interrupt request signal and the CPU is enabled to service it, the normal state sequence of the CPU is interrupted at the end of an instruction. The CPU signals the interrupt circuits through the five ROMC lines. The requesting local interrupt circuit sends a 16-bit interrupt address vector (from the interrupt address generator) onto the data bus in two consecutive bytes. The address is made available to the program counter through the address demultiplexer circuits. It is simultaneously made available to all other devices connected to the data bus and is the address of the next instruction to be executed. The program counter of each memory device is set with this new address while the stack register is loaded with the previous contents of the program counter. The information in the program counter is lost. Thus, the next instruction to be executed is determined by the value of the interrupt address vector.

The interrupt control bit (ICB) of the CPU (loaded in the W register) allows interrupts to be recognized. Clearing the ICB prevents acknowledgement of interrupts. The ICB is cleared during power-on, during external reset, and after an interrupt is acknowledged. The interrupt status of the PSU, PIO, or memory interface (MI) device is not affected by execution of the disable interrupt (DI) instruction from the CPU. At the conclusion of most instructions, the fetch logic checks the state of the interrupt request line. If an interrupt, occurs the next instruction fetch cycle is suspended and the system is forced into an interrupt sequence.

Figure 4 F8 System Interrupt Interconnection



Interrupt Sequence

Figure 5 details the interrupt sequence that occurs, whether the interrupt request is from an external source through the $\overline{EXT}$ $\overline{INT}$ pin or from the PIO device internal timer. Events are labeled A through G.

Event A
An interrupt request must satisfy a set-up time requirement. If not satisfied, the $\overline{INT}$ $\overline{REQ}$ signal delays going low until the next negative edge of the WRITE clock.

Event B
Event B represents the instruction being executed when the interrupt occurs. The last cycle of B is normally the instruction fetch for the next cycle. However, if B is not a privileged instruction and the CPU interrupt control bit is set, the last cycle becomes a freeze cycle raher than a fetch. At the end of the freeze cycle the interrupt request latches are inhibited from altering the interrupt daisy chain so that sufficient time is allowed for the daisy chain to settle.

If B is a privileged instruction, the instruction fetch is not replaced by a freeze cycle; instead, the fetch is performed and the next instruction is executed. Although unlikely to be encountered, a series of privileged instructions would be executed sequentially. One more instruction (a protected instruction) is executed after the last privileged instruction. The last cycle of the protected instruction then performs the freeze.

In figure 5, the dashed lines on the $\overline{EXT}$ $\overline{INT}$ (EI) timing illustrate the last opportunity for the $\overline{EXT}$ $\overline{INT}$ signal to cause the last cycle of a nonprotected instruction to become a freeze cycle. The freeze cycle is a short cycle (four Φ clock periods) in all cases except where B is the decrement scratchpad instruction, in which case the freeze cycle is a long cycle (six Φ clock periods).

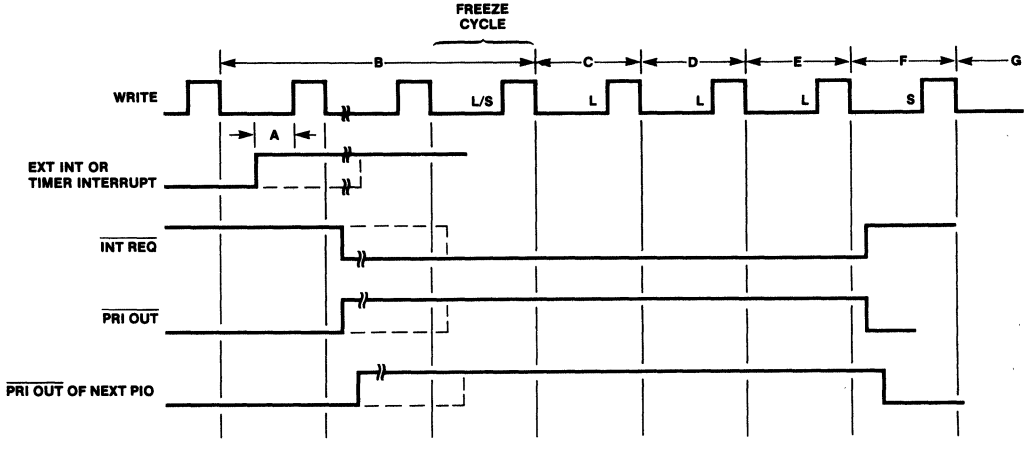
The $\overline{INT}$ $\overline{REQ}$ signal goes low on the next negative edge of the WRITE signal if both the $\overline{PRI}$ $\overline{IN}$ signal is low and the appropriate interrupt enable bit of the ICP is set.

Event C
This is a no-operation (NO-OP) long cycle, allowing time for the $\overline{PRI}$ $\overline{IN}$/$\overline{PRI}$ $\overline{OUT}$ chain to settle. At a 2-Mhz Φ clock rate, a total of seven PIO, PSU, or MI devices can be daisy-chained without the need for look-ahead logic.

Event D
In PSU circuits, the program counter (PO) is pushed to the stack register (P) to save the return address. The interrupting PIO places the lower eight bits of the interrupt vector address onto the data bus. This is always a long cycle.

Event E
In this long cycle, the PIO places the upper eight bits of the interrupt vector address onto the data bus.

Figure 5 Interrupt Sequence



Event F
In this short cycle, the PIO interrupting interrupt request latch is cleared. Also, the CPU interrupt control bit is cleared, thus disabling interrupts until an EI instruction is performed. Additionally, during Event F, the $\overline{\text{PRI IN}}/\overline{\text{PRI OUT}}$ daisy-chain freeze is removed, since the interrupt vector address has been passed to the CPU. Another action is the fetch of the instruction from the interrupt address.

Event G
This event starts executing the first instruction of the interrupt service routine.

Summary of Interrupt Sequence
For the PIO, the interrupt response time is defined as the time elapsed between the occurrence of the $\overline{\text{EXT INT}}$ signal going active (or the timer transition to H'N') and the beginning of execution of the first instruction of the interrupt service routine. The interrupt response time is a variable dependent on what the microprocessor is doing when the interrupt request occurs.

As shown in figure 5, the minimum interrupt response time is three long cycles plus two short cycles plus one write clock pulse width plus a set-up time of an $\overline{\text{EXT INT}}$ signal prior to the leading edge of the write pulse, a total of 27 ϕ clock periods plus the set-up time. At 2 MHz, this is 14.25 μs. Although the maximum could theoretically be infinite, a practical maximum is 35 μs (based on the interrupt request occurring near the beginning of a PI and LR K, P sequence).

ROMC States

Table 4 shows the function performed by the PIO device for each ROMC command. Each function is performed entirely within one machine cycle (one cycle of the WRITE clock). All other ROMC states are decoded as NO-OP.

Table 4 PIO Functions Versus ROMC States

ROMC State		
Binary	**Hex**	**PIO Functions**
01111	OF	If this circuit is interrupting and is highest in the priority chain, move lower half of interrupt vector into the data bus.
10000	10	Place interrupt circuitry in an inhibit state that prevents altering the interrupt priority chain.
10011	13	If the contents of the data bus in the prior cycle was an address of I/O ports on this device, move the current contents of the data bus into the appropriate port (I/O A, I/O B, timer or control).
11011	1B	If the contents of the data bus in the prior cycle was an address of I/O ports on this device, move the contents of the appropriate I/O port onto the data bus (I/O A or I/O B).

Timing Characteristics

Timing signals are illustrated in figures 3, 6, and 7; the signal timing characteristics are presented in table 5.

Figure 6 F3861 Input/Output Timing

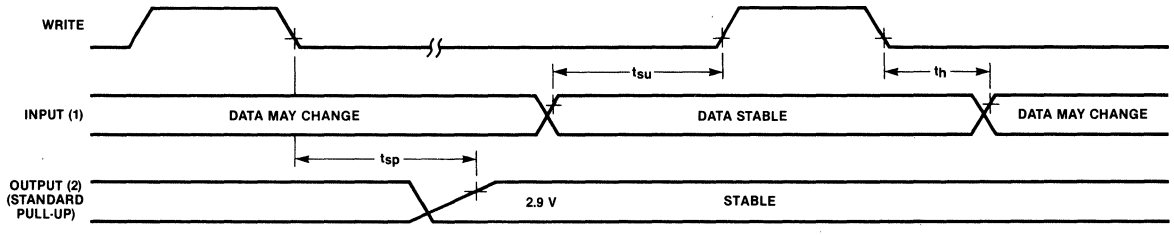
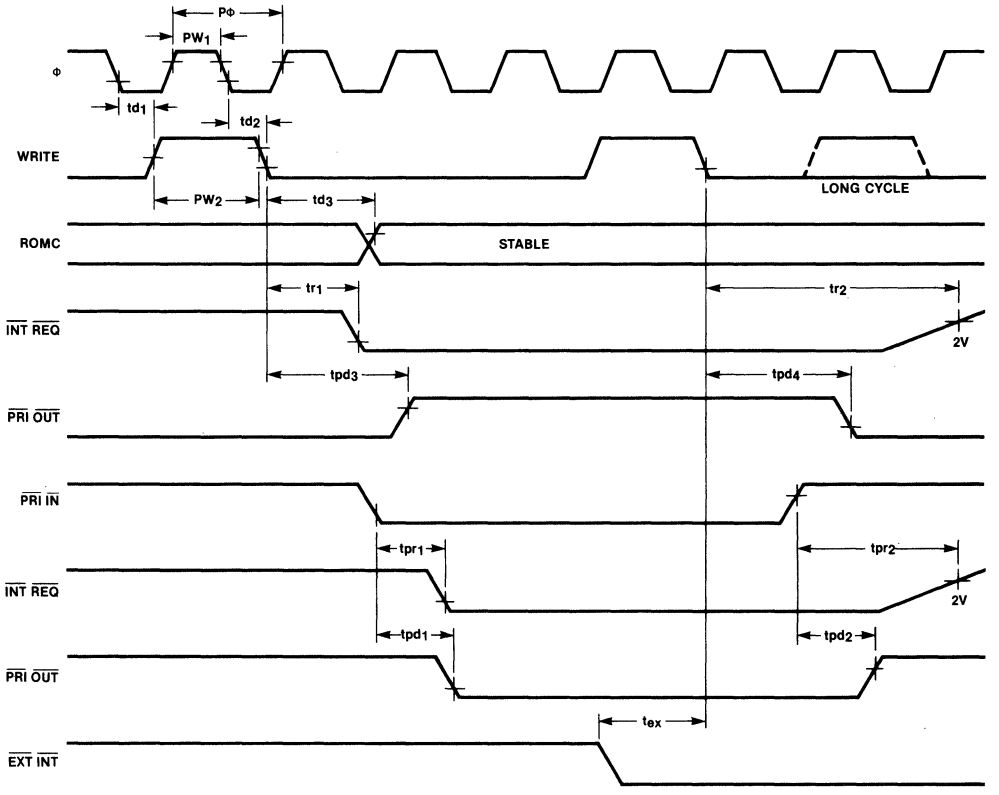


Figure 7 F3861 Interrupt Logic Timing



Note: Timing measurements are made at valid logic level to valid logic level of the signals references, unless otherwise noted.

Table 5 F3861 Timing Characteristics

The ac characteristics are $V_{SS} = 0V$, $V_{CC} = +5V(\pm 5\%)$, $T_A = 0$ C to + 70°C.

Symbol	Parameter	Min	Typ	Max	Units	Test Conditions
PΦ	Φ Period	0.5		10	5μs	
PW_1	Φ Pulse Width	180		PΦ – 180	ns	t_r, t_f = ns typ.
td_1	Φ to WRITE + Delay	60		250	ns	C_L = 100 pF
td_2	Φ to WRITE – Delay	60		225	ns	C_L = 100 pF
td_4	WRITE to DB Input Delay			2PΦ + 1.0	μs	
PW_2	WRITE Pulse Width	PΦ – 100		PΦ	ns	t_r, t_f = 50 ns typ.
PW_S	WRITE Period; Short		4PΦ			
PW_L	WRITE Period; Long		6PΦ			
td_3	WRITE to ROMC Delay WRITE to DB Output Delay			550	ns	
td_7	WRITE to $\overline{DBDR}$ - Delay	2PΦ + 100 – td_2	2PΦ + 200	2PΦ + 850 – td_2	ns	C_L = 100 pF
td_8	WRITE to $\overline{DBDR}$ + Delay		200		ns	Open Drain
tr_1	WRITE to $\overline{INT}$ $\overline{REQ}$ – Delay			430	ns	C_L = 100 pF(1)
tr_2	WRITE to $\overline{INT}$ $\overline{REQ}$ + Delay			430	ns	C_L = 100 pF (3)
tpr_1	PRI IN to $\overline{INT}$ $\overline{REQ}$ – Delay			240	ns	C_L = 100 pF (2)
tpr_2	PRI IN to $\overline{INT}$ $\overline{REQ}$ + Delay			240	ns	C_L = 100 pF
tpd_1	PRI IN to $\overline{PRI}$ $\overline{OUT}$ – Delay					
tpd_2	PRI IN to $\overline{PRI}$ $\overline{OUT}$ + Delay			365	ns	C_L = 50 pF
tpd_3	WRITE to $\overline{PRI}$ $\overline{OUT}$ + Delay			700	ns	C_L = 50 pF
tpd_4	WRITE to $\overline{PRI}$ $\overline{OUT}$ – Delay			640	ns	C_L = 50 pF
$*t_{sp}$	WRITE to Output Stable			2.5	μs	C_L = 50 pF Standard Pull-up
$*t_{su}$	I/O Setup Time	1.3			μs	
$*t_h$	I/O Hold Time	0			ns	
$*t_{ex}$	$\overline{EXT}$ $\overline{INT}$ Setup Time	400			ns	

Notes:

1. Assume Priority In was enabled ($\overline{PRI}$ $\overline{IN}$ = 0) in the previous F8 cycle before the interrupt is detected in the PIO.
2. The PSU has an interrupt pending before priority in is enabled.
3. Assume the pin is tied to the $\overline{INT}$ $\overline{REQ}$ input of the F3850 CPU.
4. The starred ★ parameters in the table represent those most frequently of importance when interfacing to an F8 system. Other parameters are typically those that are relevant between F8 chips and are not normally of concern to the user.
5. Input and output capacitance is 3 to 5 pF typical on all pins except V_{DD}, V_{GG}, and V_{SS}.

DC Characteristics

The dc characteristics of the F3861 PIO are supplied in table 6.

Table 6 F3861 PIO DC Characteristics

Symbol	Parameter	Signal	Min	Max	Units	Test Conditions
V_{IH}	Input High Voltage	Data Bus (DBO-DB7)	3.5	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
V_{OH}	Output High Voltage		3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 1.6$ mA
I_{IH}	Input High Current			1	μA	$V_{IN} = 6V$, 3-State mode
I_{OL}	Input Low Current			−1	μA	$V_{IN} = V_{SS}$,3-State mode
V_{IH}	Input High Voltage	Clock Lines	4.0	V_{DD}	V	
V_{IL}	Input Low Voltage	(Φ,WRITE)	V_{SS}	0.8	V	
I_L	Leakage Current			1	μA	$V_{IN} = 6V$
V_{IH}	Input High Voltage	Priority In and Control	3.5	V_{DD}	V	
V_{IL}	Input Low Voltage	Lines (PRI IN, ROMCO-	V_{SS}	0.8	V	
I_L	Leakage Current	ROMC4)		1	μA	$V_{IN} = 6V$
V_{OH}	Output High Voltage	Priority Out ($\overline{PRI}$ $\overline{OUT}$)	3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 100\ \mu A$
V_{OH}	Output High Voltage	Interrupt Request			V	Open Drain Output (1)
V_{OL}	Output Low Voltage	($\overline{INT}$ $\overline{REQ}$)	V_{SS}	0.4	V	$I_{OL} = 1$ mA
I_L	Leakage Current			1	μA	$V_{IN} = 6V$
V_{OH}	Output High Voltage	Data Bus Drive				External Pull-up
V_{OL}	Output Low Voltage	($\overline{DBDR}$)	V_{SS}	0.4	V	$I_{OL} = 2$ mA
I_L	Leakage Current			1	μA	$V_{IN} = 6V$
V_{IH}	Input High Voltage	External Interrupt	3.5		V	
V_{IL}	Input Low Voltage	($\overline{EXT}$ $\overline{INT}$)		1.2	V	
V_{IC}	Input Clamp Voltage			15	V	$I_{IH} = 185\ \mu A$
I_{IH}	Input High Current			10	μ	$V_{IN} = V_{DD}$
I_{IL}	Input Low Voltage			−225	μA	$V_{IN} = 2V$
I_{IL}	Input Low Current		−150	−500	μA	$V_{IN} = V_{SS}$
V_{OH}	Output High Voltage	I/O Port	3.9	V_{DD}	V	$I_{OH} = -30\ \mu A$
V_{OH}	Output High Voltage	(Standard Pull-Up)	2.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 2$mA
V_{IH}	Input High Voltage		2.9	V_{DD}	V	Internal Pull-up to V_{DD} (3)
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_{IL}	Leakage Current, DC/PC			−1.6	mA	$V_{IN} = 0.4$ V
I_{IL}	Leakage Current, DC/PL/DM			−2.0	mA	$V_{IN} = 0.4$ V
I_L	Input Low Current			−1.6	mA	V_{IN} V_{SS} (4) (7)

Notes:

1. Pull-up resistor to V_{DD} on CPU.
2. Positive current is defined as conventional current flowing into the pin referenced.
3. Hysteresis input circuit provides additional 0.3 V noise immunity while internal/external pull-up provides TTL compatibility.
4. Measured an K a high-level I/O port OUT port.
5. V_{SS} = °V, $V_{DD} = \pm 5V \pm 5\%$, $V_{GG} = +12V \pm 5\%$, $T_A = 0°C$ to 70°C
6. Output device off.
7. −2.0 mA for extended temperature range.

The supply currents are given in table 7.

Table 7 Supply Currents

Symbol	Parameter	Min	Typ	Max	Units	Test Conditions
I_{DD}	V_{DD} Current		30	70	mA	F = 2 MHz, Outputs Unloaded
I_{GG}	V_{GG} Current		10	18	mA	f = 2 MHZ, Outputs Unloaded

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

V_{GG}	+15 V, −0.3 V
V_{DD}	+17 V, −0.3 V
External Interrupt Input	−600 μA, +225 μA
All Other Inputs and Outputs	+7 V, −0.3 V
Storage Temperature	−55°C, +150° C
Operating Temperature	0°C, +70°C

Note
All voltages are with respect to V_{SS}.

Recommended Operating Ranges

Part Number	Supply Voltage (V_{DD})			V_{GG}			
	Min	Typ	Max	Min	Typ	Max	V_{SS}
F3861	+4.75 V	+5 V	+ 5.25 V	+11.4 V	+12 V	+12.6 V	0 V

Ordering Information

Part Number	Package	Temperature Range
F3861	Ceramic	C
F3861 DM	Ceramic	M
*F3861 PC	Plastic	C

C = Commericial Temperature Range 0° to +70°C
L = Limited Temperature Range −40°C to +85°C
M = Military Temperature Range −55°C to +125°C
* Version A, B, C, D, and E are stocked items.

FAIRCHILD
A Schlumberger Company

F3871
Peripheral Input/Output

Microprocessor Product

Description

The Fairchild F3871 Peripheral Input/Output (PIO) device provides two 8-bit I/O ports, external interrupt, and a programmable timer. An 8-bit-wide bidirectional data bus transfers I/O data bytes between the F3870 Central Processing Unit (CPU) and the PIO.

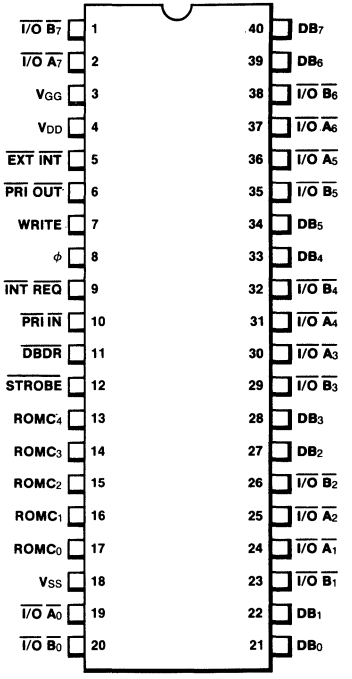
The PIO is used in systems that require the I/O capability and interrupt functions of the F3851 PSU, but do not need the read-only memory (ROM) storage of the PSU. The PIO is pin-compatible with the PSU.

The F3871 has the same improved timer and ready strobe output as the F3870 CPU; therefore, for software compatibility with the F3870, the F3871 PIO should be used in the F8 multichip configurations.

The F3871 is manufactured using isoplanar N-channel silicon-gate technology; therefore, power dissipation is very low (less than 200 mW).

- **16 Bidirectional, Individually Controlled I/O Lines Organized as Two 8-Bit Ports**
- **I/O Strobe**
- **Programmable Timer — Preset, Start, Stop, and Read-Back Ability: Selectable Timer Count Rates**
- **Full Interrupt Level — Daisy-Chain Expandable, Independent Interrupt Address Vectors for Timer and External Interrupt**
- **Pulse Width Measurement Capability**
- **+5 V and +12 V Power Supplies**
- **2 MHz Operation**
- **TTL and LSTTL Compatible**
- **Low Power Dissipation, Typically Less Than 200 mW**

Connection Diagram



3

Signal Functions

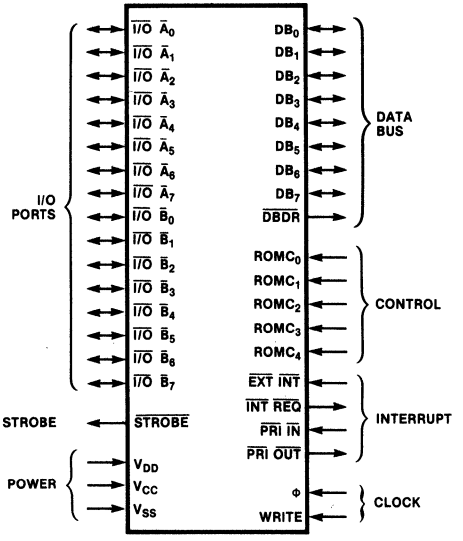


Figure 1 F3871 Block Diagram

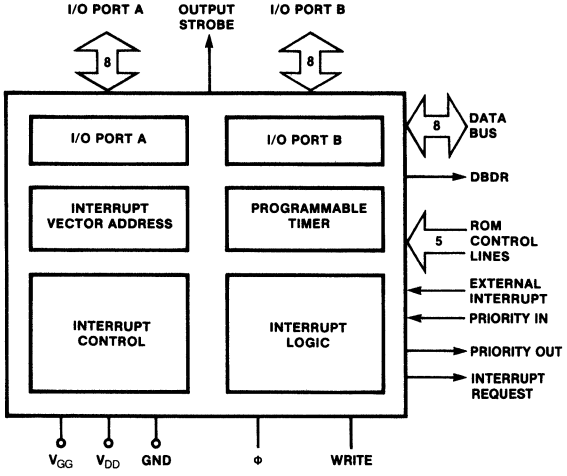


Table 1 F3871 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
ϕ	8	Clock	The two clock input signals that originate at the F3850 CPU.
WRITE	7		
I/O Ports			
$\overline{I/O\ A_0}$-$\overline{I/O\ A_7}$	19, 24, 25, 30, 31, 36, 37, 2	I/O Ports A	Bidirectional ports through which the PIO communicates with logic external to the microprocessor system.
$\overline{I/O\ B_0}$-$\overline{I/O\ B_7}$	20, 23, 26, 29, 31, 35, 38, 1	I/O Ports B	
Control			
$ROMC_0$-$ROMC_4$	17, 16, 15, 14, 13	Read-Only Memory Control	Input signals that originate at the F3850 CPU control internal functions of the PIO.
Data Bus			
DB_0-DB_7	21, 22, 27, 28, 33, 34, 39, 40	Data Bus	Bidirectional 3-state lines that link the PIO to all other devices within the microprocessor system.
$\overline{DBDR}$	11	Data Bus Drive	A low output, open drain signal that indicates the data bus currently contains data flowing from the PIO.
Strobe			
$\overline{STROBE}$	12	Strobe	Provides a single low output pulse after valid data is present on $\overline{I/O\ A_0}$-$\overline{I/O\ A_7}$ during an output instruction.
Interrupt			
EXT INT	5	External Interrupt	A high-to-low transition on this input signal is interpreted as an interrupt request from an external device.
$\overline{INT\ REQ}$	9	Interrupt Request	This output signal is the $\overline{INT\ REQ}$ input to the F3850 CPU; it must be output low to interrupt the CPU, which occurs only if $\overline{PRI\ IN}$ is low and PIO interrupt control logic is requesting an interrupt.
$\overline{PRI\ IN}$	10	Priority In	Unless this input signal is low, the PIO will not set the $\overline{INT\ REQ}$ signal low in response to an interrupt.
$\overline{PRI\ OUT}$	6	Priority Out	This output signal becomes the $\overline{PRI\ IN}$ signal to the next device in the interrupt-priority daisy chain; it is output high unless the $\overline{PRI\ IN}$ signal is entering the PIO low and the PIO is not requesting an interrupt.
Power			
V_{DD}	4	Power Supply	+5 V (±5%)
V_{GG}	3	Power Supply	+12 V (±5%)
V_{SS}	18	Ground	System ground—0 V; V_{DD} and V_{GG} are referenced to V_{SS}.

Device Organization

The peripheral input/output device includes I/O logic, timer logic, interrupt logic, data bus logic, and control logic, as illustrated in figure 1.

The interrupt logic responds to an interrupt request signal originating from internal timer logic or an external device. Based on priority considerations, the interrupt request is passed on to the F3850 CPU.

The programmable timer uses a polynomial shift register in conjunction with interrupt logic to generate real-time intervals.

The 8-bit data bus in the PIO is the main path for transfer of information between the F3850 CPU and other devices in the F8 microprocessor system. The device has four pre-assigned I/O port addresses: the lowest two are assigned to the two I/O ports, A and B, and are used to transfer data to and from external devices. The other two I/O addresses are assigned to two internal registers of the PIO that control interrupt logic and are treated as I/O ports.

Signal Descriptions

The F3871 input and output signals are described in table 1.

System Clock Timing

All timing within the PIO is controlled by the ϕ and WRITE signals input from the F3850 CPU. (Refer to the F3850 data sheet for a description of these clock signals.) The WRITE clock refreshes and updates PIO registers, which are dynamic. The ϕ clock drives sequencing logic to precharge interrupt logic. The ϕ clock also drives the programmable timer.

I/O Ports

The PIO has two bidirectional 8-bit I/O ports used to transmit data between it and external devices. In binary notation, the address for port A is XXXXXX00 and for port B is XXXX-XX01, where the X binary digits are the unique I/O port select code for the PIO (see table 2). For example, if the port select code is 000001, port A can be called port 4 and port B can be called port 5. (The PIO port select code is never designated as all "0"s, since ports 0 and 1 are reserved for the F3850 CPU.) In addition, the interrupt control port (ICP) is addressed as port XXXXXX10 and the binary timer is addressed as port XXXXXX11, which become ports 6 and 7, respectively, for the port select code example just given.

Table 2 F3871 PIO Port Addresses

Address	Assigned To
XXXX XX00	I/O Port A
XXXX XX01	I/O Port B
XXXX XX10	Interrupt Control Register
XXXX XX11	Programmable Timer

The port and interrupt address vector assignments for the F3871 are given in table 3.

Table 3 F3871 Port and Address Assignments (HEX)

Version	Port Addresses	Port Output Type	Interrupt Address Vector	
			Timer	External
3871E	4-7	Standard	0020	00A0
3871F	4-7	Direct Drive	0020	00A0
3871G	4-7	Open Drain	0020	00A0
3871H	8-B	Standard	4420	44A0

Port Pin Description

An output instruction (OUT or OUTS) causes the contents of the CPU accumulator (ACC) to be latched into the addressed port. An input instruction (IN or INS) transfers the contents of the port to the ACC (port 6 is an exception that is described in the "Timer and Interrupt Control Port" section). The I/O pins on the PIO are logically inverted; the schematic of an I/O pin and available output drive options are shown in figure 2. Each output pin has an output latch that holds the data last output to that pin. The I/O ports of the PIO are configured in the standard pull-up option.

The $\overline{\text{STROBE}}$ output is always configured in a manner similar to a standard output, except that it is capable of driving three TTL loads.

Each I/O port pin is a wire-AND structure between an internal output data latch and the external signal. The latch is loaded from the data bus. The output latches are not initialized by the system reset sequence.

When transmitting data through an I/O port, the pin can be connected directly to a TTL gate input; data is input to the pin from a TTL device output. Since the I/O pin and the TTL device output are wire-ANDed, it is possible for the state of one to affect the transfer of data out from the I/O pin or in from the TTL device output. In most cases, I/O port bits should, therefore, be set for a high level (logic 0), before data input, to prevent incoming logic "0"s from being masked by logic "1"s present at the port from previous outputs. However, the ability to mask bits of a port to logic 1 is useful during some input functions.

Figure 2 F3871 I/O Pin Diagram with Output Buffer Options

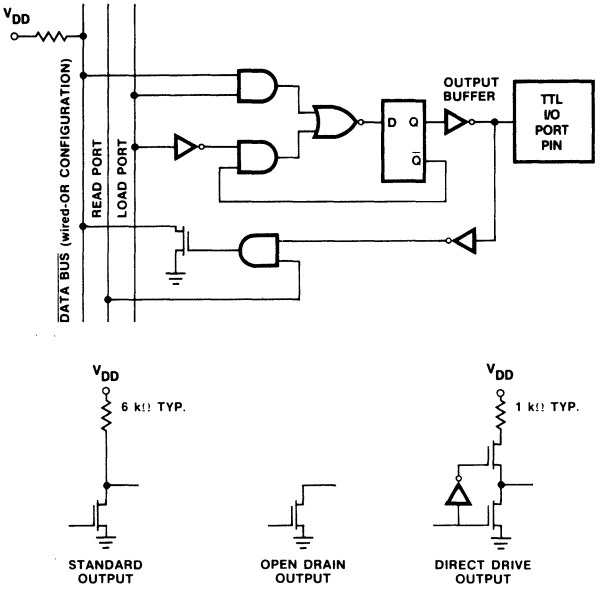
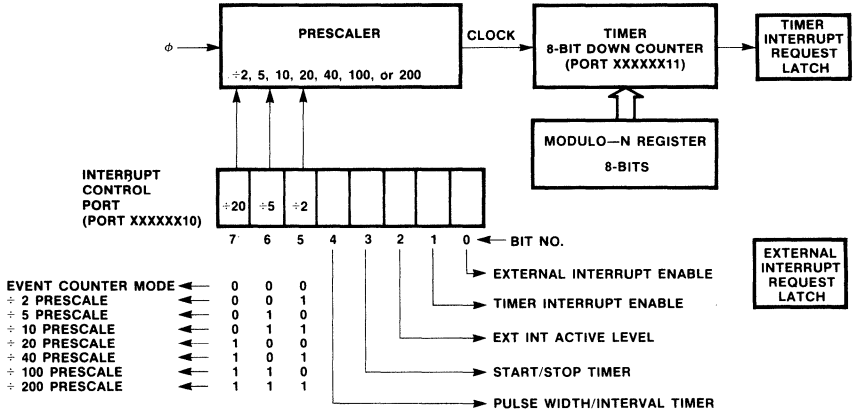


Figure 3 F3871 Timer and Interrupt Control Port Block Diagram



Strobe
An output ready strobe is associated with port A. This flag is used to signal a peripheral device that the F3871 has just completed an output of new data to port A. Since the strobe provides a single low pulse shortly after the output operation is completed, either edge can be used to signal the peripheral. The $\overline{\text{STROBE}}$ signal is also used as an input strobe by performing a dummy output of H'00' to port A after completing the input operation.

Timer and Interrupt Control Port
The timer is software-programmable to operate in one of three modes: the interval timer mode, the pulse width measurement mode, and the event counter mode. As shown in figure 3, an 8-bit register (interrupt control port), a programmable prescaler, and an 8-bit modulo-N register are associated with the timer.

The desired timer mode, prescale value, timer start and stop, active level of the external interrupt pin, and local interrupt enable or disable are selected by the proper bit configuration output from the accumulator to the interrupt control port (port XXXXXX10), with an OUT or OUTS instruction.

Interrupt Control Port
A special situation exists when reading the ICP with an IN or INS instruction. The accumulator is not loaded with the contents of the ICP; instead, accumulator bits 0 through 6 are loaded with "0"s, while bit 7 is loaded with the logic level being applied to the EXT INT pin, thus determining the status of the EXT INT signal without servicing an external interrupt request. This capability is useful in two ways: establishing a high-speed polled handshake procedure and using the EXT INT pin as an extra input pin if external interrupts are not required and if the timer is used only in the interval timer mode. However, if it is desirable to read the contents of the ICP, one of the 64 scratchpad registers is used to save a copy of material written to the ICP.

The timer clock rate in the interval timer mode is determined by the frequency of the ϕ clock and by the division value selected for the prescaler. If ICP bit 5 is set and bits 6 and 7 are cleared, the prescaler divides ϕ by two. If bit 6 or 7 is individually set, the prescaler divides ϕ by five or twenty, respectively. Combinations of bits 5, 6, and 7 can also be selected. For example, if bits 5 and 7 are set while 6 is cleared, the prescaler divides by 40. Thus, possible prescaler values are $\div 2$, $\div 5$, $\div 10$, $\div 20$, $\div 40$, $\div 100$, and $\div 200$.

Any of three conditions causes the prescaler to reset:
1. When the timer is stopped by clearing ICP bit 3.

2. When an output instruction to the timer (port XXXXXX11) is executed.

3. On the trailing edge transition of the EXT INT pin, when in the pulse width measurement mode.

An OUT or OUTS instruction to the timer loads the contents of the accumulator (the interval value) to both the timer and the 8-bit modulo-N register, resets the prescaler, and clears any previously stored timer interrupt request. The timer is clocked by the prescaler in the interval timer mode and in the pulse width modulator mode; the prescaler is not used in the event counter mode. The modulo-N register is a buffer that saves the value most recently output to port XXXXXX11 and is used in all three timer modes.

Interval Timer Mode
When ICP bit 4 is cleared (logic 0) and at least one prescale bit is set, the timer operates in the interval timer mode. When bit 3 of the ICP is set, the timer starts counting down from the modulo-N value. After counting down to H'01', the timer returns to the modulo-N value at the next count. On the transition from H'01' to H'N', the timer sets a timer interrupt request latch. Note that the interrupt request latch is set by the transition of H'N' in the timer, thus allowing a full 256 counts if the modulo-N register is preset to H'00'.

If bit 1 of the ICP is set and the $\overline{\text{PRI IN}}$ signal is low, the interrupt request is passed to the F3850 CPU. However, if bit 1 of the ICP is a logic 0, the interrupt request is not passed on to the CPU. If bit 1 is subsequently set, the interrupt request is then passed to the CPU. Only two events reset the timer interrupt request latch:
1. Acknowledgement by the CPU of the timer interrupt request.

2. Performance of a new load operation of the modulo-N register.

(The interrupt priority sequence is discussed in the "Interrupt Sequence" section.)

For example, if the modulo-N register is loaded with H'64' (decimal 100), the timer interrupt request latch is set at the 100th count following the timer start, and the timer interrupt request latch is repeatedly set on precise 100-count intervals. If the prescaler is set at $\div 40$, the timer interrupt request latch is set every 4000 ϕ clock periods. For a 2 MHz ϕ clock, this setting produces 2 ms intervals.

The range of possible intervals is from 2 to 51,200 ϕ clock periods (1 μs to 25.6 ms for a 2 MHz ϕ clock). However, approximately 50 ϕ clock periods is a practical minimum, because the time between setting the interrupt request latch and the execution of the first instruction of the interrupt service routine is at least 29 ϕ clock periods (the response time is dependent on how many privileged instructions are encountered when the request occurs). To establish time

intervals greater than 51,200 ϕ clock periods, use the timer interrupt service routine to count the number of interrupts, saving the result in one or more of the scratchpad registers until the desired interval is achieved. With this technique, virtually any time interval, or several time intervals, can be generated.

The F3871 timer can be read at any time and in any mode, using an input instruction (IN or INS), and can take place "on-the-fly" without interfering with normal timer operation. The timer can be stopped at any time by clearing bit 3 of the ICP. The timer holds its current contents indefinitely and resumes counting when bit 3 is set again. The prescaler is reset whenever the timer is stopped; thus, a series of starting and stopping results in a cumulative truncation error.

For a free-running timer in the interval timer mode, the time interval between any two interrupt requests can be in error by plus or minus six ϕ clock periods, although the cumulative error over many intervals is zero. The prescaler and timer generate precise intervals for setting the timer interrupt request latch, but the time-out can occur at any time within a machine cycle. (There are two types of machine cycles: short cycles that consist of four ϕ clock periods and long cycles that consist of six ϕ clock periods.) The write clock corresponds to a machine cycle. Interrupt requests are synchronized with the write clock, thus creating the possible plus or minus six ϕ error. Additional errors can arise if the interrupt request occurs while a privileged instruction or multicycle instruction is being executed. However, for most applications, all of the above errors are negligible, especially if the desired time interval is greater than one ms. Other timer errors are summarized in the "Timing Characteristics" section.

Pulse Width Measurement Mode
When ICP bit 4 is set (logic 1) and at least one prescale bit is set, the timer operates in the pulse width measurement mode. This mode is used to accurately measure a pulse duration applied to the EXT INT pin. The timer is stopped and the prescaler is reset whenever the EXT INT pin is at its inactive level. The active level of the EXT INT pin is defined by ICP bit 2: if cleared, the EXT INT pin is active low; if set, the EXT INT pin is active high. If ICP bit 3 is set, the prescaler and timer start counting when the EXT INT signal goes through a transition to the active level.

When the EXT INT pin returns to the inactive level, the timer stops, the prescaler resets, and, if ICP bit 0 is set, an external interrupt request latch is set. (Unlike timer interrupts, external interrupts are not latched if the ICP interrupt enable is not set.)

As in the interval timer mode, the timer can be read at any time and can be stopped at any time by clearing ICP bit 3, the prescaler, and ICP bit 1 functions as previously described. The timer still functions as an 8-bit binary down counter with the timer interrupt request latch set on the timer's transition from H'01' to H'N'. Note that the EXT INT pin is not involved with loading the timer; its action is that of automatically starting and stopping the timer and of generating external interrupts. Pulse widths longer than the prescale value times the modulo-N value are easily measured by using the timer interrupt service routine to store the number of timer interrupts in one or more scatchpad registers.

The actual pulse duration is typically slightly longer than the measured value, because the status of the prescaler is not readable and is reset when the timer is stopped. Thus, for maximum accuracy, using a small division setting for the prescaler is advisable.

Event Counter Mode
When ICP bit 4 is cleared and all prescale bits (ICP bits 5, 6, and 7) are cleared, the timer operates in the event counter mode. This mode is used for counting pulses applied to the EXT INT pin. If ICP bit 3 is set, the timer decrements on each transition from the inactive level to the active level of the EXT INT pin. The prescaler is not used in this mode. As in the other two timer modes, the timer can be read at any time and can be stopped at any time by clearing ICP bit 3. ICP bit 1 functions as previously described, and the timer interrupt request latch is set on the timer's transition from H'01' to H'N'.

Normally, ICP bit 0 is kept cleared in the event counter mode; otherwise, external interrupts are generated on the transition from the inactive level to the active level of the EXT INT pin.

For the event counter mode, the minimum pulse width required on EXT INT is two ϕ clock periods, and the minimum inactive time is two ϕ clock periods; the maximum repetition rate is 500 Hz.

External Interrupts
When the timer is in the interval timer mode, the EXT INT pin is available for non-timer-related interrupts. If ICP bit 0 is set, an external interrupt request latch is set when there is a transition from the inactive level to the active level of the EXT INT pin. (The EXT INT pin is an edge-triggered input.) The interrupt request is latched either until acknowledged by the CPU or until ICP bit 0 is cleared (unlike timer interrupt requests that remain latched even when ICP bit 1 is cleared).

External interrupts are handled in the same manner when the timer is in the pulse width measurement mode or in the event counter mode, except that in the pulse width measurement mode only, the external interrupt request latch is set on the trailing edge of the EXT INT signal (i.e., on the transition from the active level to the inactive level).

Instruction Execution

The PIO responds to signals that are output by the F3850 CPU in the course of implementing instruction cycles. The PIO places data on the data bus, even in the worse case, in time for the setup required by any F3850 CPU destination. The PIO receives a byte input from the data bus when commanded by an output instruction to load one of its two I/O ports or internal registers.

The data bus drive signal ($\overline{DBDR}$) is low while data output by the PIO is stable on the data bus. Thus, a $\overline{DBDR}$ low signal indicates that the data bus currently contains data flowing from a PIO. For systems with more than one program storage unit (PSU) or PIO, the $\overline{DBDR}$ output signal can be wire-ORed and the result used as a bus data flow direction indicator. The $\overline{DBDR}$ signal can remain low until timing interval td_8 of the next instruction cycle following the one in which $\overline{DBDR}$ was set low.

The PIO device executes the OUT instruction in the same manner as it does the OUTS instruction; the same is true for the IN and INS instructions. The difference between the long- and short-form instructions is found only in the source of the I/O address.

The F8 input/output instructions place the I/O port address on the data bus during one instruction cycle and then use the data bus in the following instruction cycle to do the actual I/O data movement. The Read-Only Memory Control (ROMC) lines coming from the F3850 CPU signal the PIO that an I/O data movement is occurring during the current instruction cycle. Therefore, the PIO needs to recognize whether the contents of the data bus during the instruction cycles just prior matches any of its four assigned I/O addresses wherever the ROMC lines indicate an I/O transfer. The address select logic constantly monitors the data bus for a match to any of the four addresses and holds the information of a match through the following cycle.

Input instructions that select a port cause the contents of the selected port to be placed on the data bus during the input cycle. Only the two I/O ports (lowest two addresses) respond to input instructions. Output instructions that select a port transfer the contents of the data bus to that port. Outputs of the latches change at the end of the I/O transfer cycle.

Interrupt Handling

A typical F8 system interrupt interconnection is shown in figure 4. Each PSU and PIO has a $\overline{PRI\ IN}$ and a $\overline{PRI\ OUT}$ line so that they can be daisy-chained together in any order to form a priority level of interrupts. When a PIO receives an interrupt (either timer or external), it pulls its $\overline{PRI\ OUT}$ output signal high, signaling all lower priority peripherals that it has a higher priority interrupt request pending on the CPU. When the PIO device's $\overline{PRI\ IN}$ input signal is pulled high by a higher priority peripheral, signaling the PIO that there is a still higher priority interrupt request pending, it passes that signal along by pulling its $\overline{PRI\ OUT}$ signal high. When the CPU processes an interrupt request, it commands the interrupting device to place its interrupt vector address on the data bus. Only the device with a $\overline{PRI\ IN}$ signal low and an interrupt request pending responds. Should there be another lower priority device with a pending request, it does not respond at that time because its $\overline{PRI\ IN}$ input signal is high.

If there is both a timer interrupt request and an external interrupt request when the CPU starts to process the requests, the timer interrupt is handled first.

Within each local interrupt control circuit is a 16-bit interrupt address vector. This vector is the address to which the program counter is set after an interrupt is acknowledged and is, therefore, the address of the first executable instruction of the interrupt routine.

The interrupt address is unique to the version of the PIO device selected by the user. Fifteen bits are fixed: bits 0 through 6 and bits 8 through 15. Bit 7 (2^7) is dependent on the type of interrupt. This bit is a 0 for internal timer-generated interrupts and a 1 for external interrupts.

When the interrupt logic sends an interrupt request signal and the CPU is enabled to service it, the normal state sequence of the CPU is interrupted at the end of an instruction. The CPU signals the interrupt circuits through the five ROMC lines. The requesting local interrupt circuit sends a 16-bit interrupt address vector (from the interrupt address generator) onto the data bus in two consecutive bytes.

The address is made available to the program counter through the address demultiplexer circuits. It is simultaneously made available to all other devices connected to the data bus. It is the address of the next instruction to be executed. The program counter of each memory device is set with this new address while the stack register is loaded with the previous contents of the program counter. The information in the program counter is lost. Thus, the next instruction to be executed is determined by the value of the interrupt address vector.

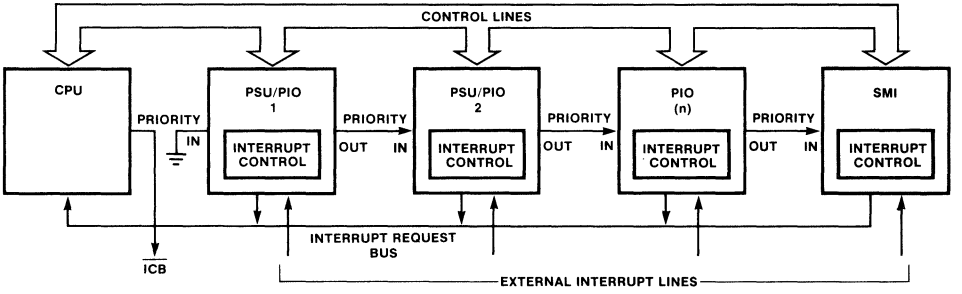
The interrupt control bit (ICB) of the CPU (loaded in the W register) allows interrupts to be recognized. Clearing the ICB prevents acknowledgement of interrupts. The ICB is cleared during power-on and external reset, and after an interrupt is acknowledged. The interrupt status of the PSU, PIO, or memory interface (MI) devices is not affected by execution of the disable interrupt (DI) instruction. At the conclusion of most instructions, the fetch logic checks the state of the interrupt request line. If there is an interrupt, the next instruction fetch cycle is suspended and the system is forced into an interrupt sequence.

The CPU allows interrupts after all F8 instructions except the following:

(PK)	PUSH K
(PI)	PUSH IMMEDIATE
(POP)	POP
(JMP)	JUMP
(OUTS)	OUTPUT SHORT (Excluding OUTS 00 and 01)
(OUT)	OUTPUT
(EI)	SET ICB
(LR W, J)	LOAD THE STATUS REGISTER FROM SCRATCHPAD
POWER ON	

As a result, it is possible to perform one more instruction after each of the above CPU instructions without being interrupted.

Figure 4 F8 System Interrupt Interconnection



Interrupt Sequence
Figure 5 details the interrupt sequence that occurs whether the interrupt request is from an external source through the EXT INT pin or from the PIO device's internal timer. The events in the sequence are labeled A through G.

Event A
An interrupt request must satisfy a set-up time requirement. If not satisfied, the $\overline{\text{INT REQ}}$ signal delays going low until the next negative edge of the write clock.

Event B
Event B represents the instruction being executed when the interrupt occurs. The last cycle of B is normally the instruction fetch for the next cycle. However, if B is not a privileged instruction and the CPU interrupt control bit is set, the last cycle becomes a freeze cycle rather than a fetch. At the end of the freeze cycle, the interrupt request latches are inhibited from altering the interrupt daisy chain so that sufficient time is allowed for the daisy chain to settle.

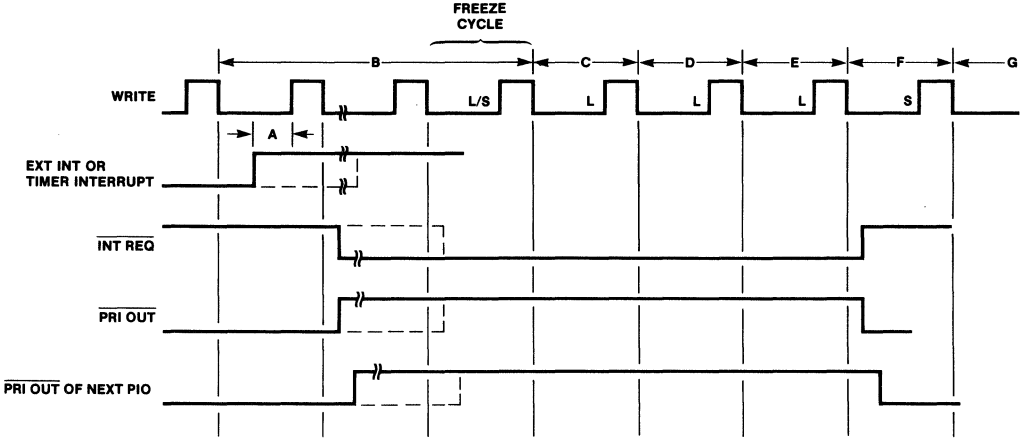
If B is a privileged instruction, the instruction fetch is not replaced by a freeze cycle; instead, the fetch is performed and the next instruction is executed. Although unlikely to be encountered, a series of priviledged instructions would be executed sequentially. One more instruction (a protected instruction) is executed after the last priviledged instruction. The last cycle of the protected instruction then performs the freeze.

The dashed lines on the EXT INT timing in figure 5, illustrate the last opportunity for the EXT INT signal to cause the last cycle of a nonprotected instruction to become a freeze cycle. The freeze cycle is a short cycle (four ϕ clock periods) in all cases except where B is the decrement scratchpad instruction, in which case the freeze cycle is a long cycle (six ϕ clock periods).

The $\overline{\text{INT REQ}}$ signal goes low on the next negative edge of WRITE if both the $\overline{\text{PRI IN}}$ signal is low and the appropriate interrupt enable bit of the ICP is set.

Event C
This is a no-operation (NO-OP) long cycle, allowing time for the $\overline{\text{PRI IN}}/\overline{\text{PRI OUT}}$ chain to settle. At a 2MHz ϕ clock rate, a total of seven PIO, PSU, or MI devices can be daisy-chained without the need for look-ahead logic.

Figure 5 F3871 Interrupt Sequence



Event D
In PSU circuits, the program counter (PO) is pushed to the stack register (P) to save the return address. The interrupting PIO places the lower eight bits of the interrupt vector address onto the data bus. This is always a long cycle.

Event E
In this long cycle, the PIO places the upper eight bits of the interrupt vector address onto the data bus.

Event F
In this short cycle, the PIO interrupting interrupt request latch is cleared. Also, the CPU interrupt control bit is cleared, thus disabling interrupts until an EXT INT instruction is performed. Additionally, during Event F, the $\overline{\text{PRI}}$ $\overline{\text{IN/PRI OUT}}$ daisy-chain freeze is removed, since the interrupt vector address has been passed to the CPU. Another action is the fetch of the instruction from the interrupt address.

Event G
This event starts executing the first instruction of the interrupt service routine.

Summary of Interrupt Sequence
For the PIO, the interrupt response time is defined as the time elapsed between the occurrence of the EXT INT signal going active (or the timer transition to H'N') and the beginning of execution of the first instruction of the interrupt service routine. The interrupt response time is a variable dependent on what the microprocessor is doing when the interrupt request occurs.

As shown in figure 5, the minimum interrupt response time is three long cycles plus two short cycles plus one WRITE clock pulse width plus a setup time of an EXT INT signal before the leading edge of the WRITE pulse—a total of 27 ϕ clock periods plus the setup time. At 2 MHz, this is 14.25 μs. Although the maximum could theoretically be infinite, a practical maximum is 35 μs (based on the interrupt request occurring near the beginning of a PI and LR K, P sequence).

ROMC States
Table 4 shows the function performed by the PIO device for each ROMC command. Each function is performed entirely within one machine cycle (one cycle of the write clock). All other ROMC states are decoded as NO-OP.

Table 4 PIO Functions Versus ROMC States

ROMC State		
Binary	**Hex**	**PIO Functions**
01000	08	Reset command. Load port A, port B, timer, and interrupt control port with H'00'.
01111	0F	If this circuit is interrupting and is highest in the priority chain, move lower half of interrupt vector into the data bus.
10000	10	Place interrupt circuitry in an inhibit state that prevents altering the interrupt priority chain.
10011	13	If this circuit is interrupting and is highest in the priority chain, move upper half of interrupt vector into the data bus and reset the interrupt circuit.
11010	1A	If the contents of the data bus in the prior cycle was an address of I/O ports on this device, move the current contents of the data bus into the appropriate port (I/O A, I/O B, timer, or control).
11011	1B	If the contents of the data bus in the prior cycle was an address of I/O ports on this device, move the contents of the appropriate I/O port onto the data bus (I/O A or I/O B).

Timing Characteristics

Timing signals are illustrated in figures 6 through 10, and the signal characteristics are presented in tables 5 through 9. Definitions for the timing characteristics are as follows:

Error = Indicated Time Value — Actual Time Value

$t_{psc} = t\phi \times$ Prescale Value

Interval Timer Mode

Single interval error, free running (Note 3) ..... $\pm 6t\phi$
Cumulative interval error, free running (Note 3) 0
Error between two timer reads (Note 2) ..... $\pm(t_{psc} + t\phi)$
Start timer to stop timer error
(Notes 1, 4) ..... $+t\phi$ to $-(t_{psc} + t\phi)$
Start timer to read timer error
(Notes 1, 2) ..... $-5t\phi$ to $-(t_{psc} + 7t\phi)$
Start timer to interrupt request error
(Notes 1, 3) ..... $-2t\phi$ to $-8t\phi$
Load timer to stop timer error (Note 1) .... $+t\phi$ to $-(t_{psc} + 2t\phi)$
Load timer to read timer error
(Notes 1, 2) ..... $-5t\phi$ to $-(t_{psc} + 18t\phi)$
Load timer to interrupt request error
(Notes 1, 3) ..... $-2t\phi$ to $-9t\phi$

Pulse Width Measurement Mode

Measurement accuracy (Note 4) ..... $+t\phi$ to $-(t_{psc} + 2t\phi)$
Minimum pulse width of EXT INT pin ..... $2t\phi$

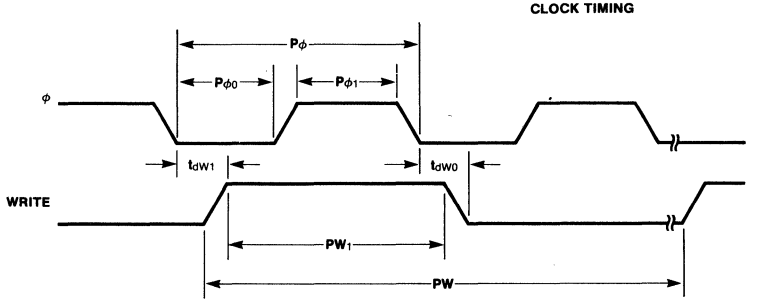
Event Counter Mode

Minimum active time of the EXT INT pin ..... $2t\phi$
Minimum inactive time of the EXT INT pin ..... $2t\phi$

NOTES

1. All times that entail loading, starting, or stopping the timer are referenced from the end of the last machine cycle of the OUT or OUTS instruction.
2. All times that entail reading the timer are referenced from the end of the last machine cycle of the IN or INS instruction.
3. All times that entail the generation of an interrupt request are referenced from the start of the machine cycle in which the appropriate interrupt request latch is set. Additional time elapses if the interrupt request occurs during a privileged or multicycle instruction.
4. Error can be cumulative if operation is repetitively performed.

Figure 6 F3871 Clock Timing



TIMING
ALL TIMING SPECIFIED AT V_{SS} = 0 V, V_{DD} = +5 V ±5%, V_{GG} = +12 V ± 5%

Figure 7 F3871 Strobe Timing

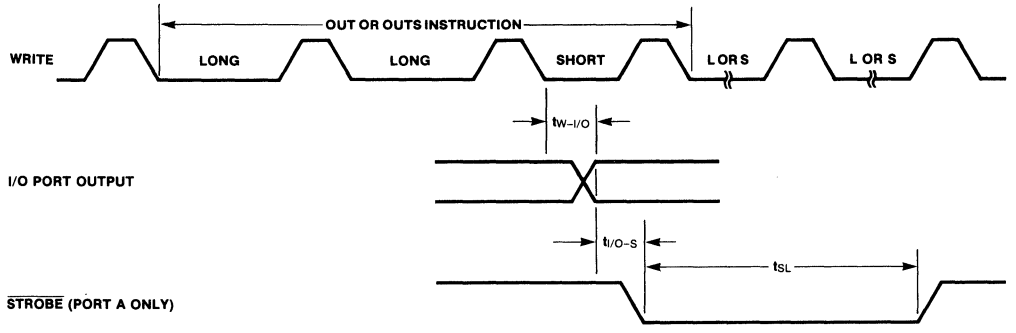


Figure 8 F3871 Input Timing

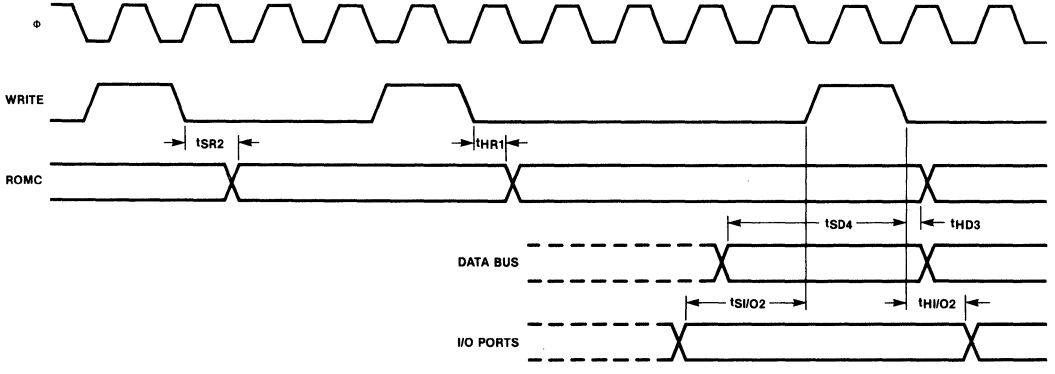


Figure 9 F3871 Output Timing

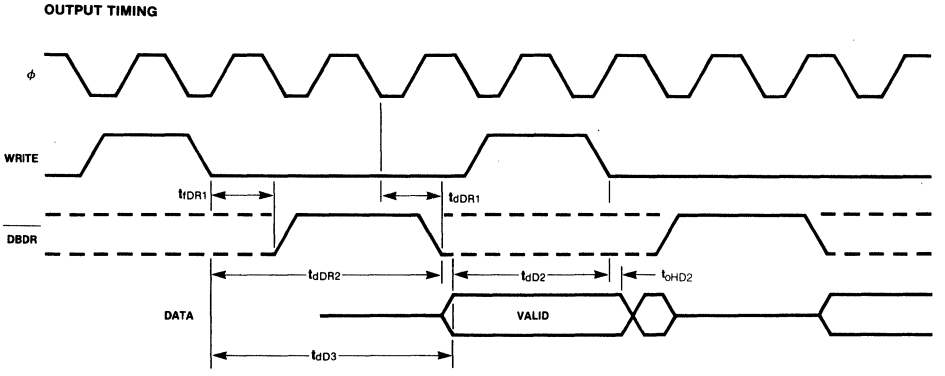


Figure 10 F3871 Interrupt Timing

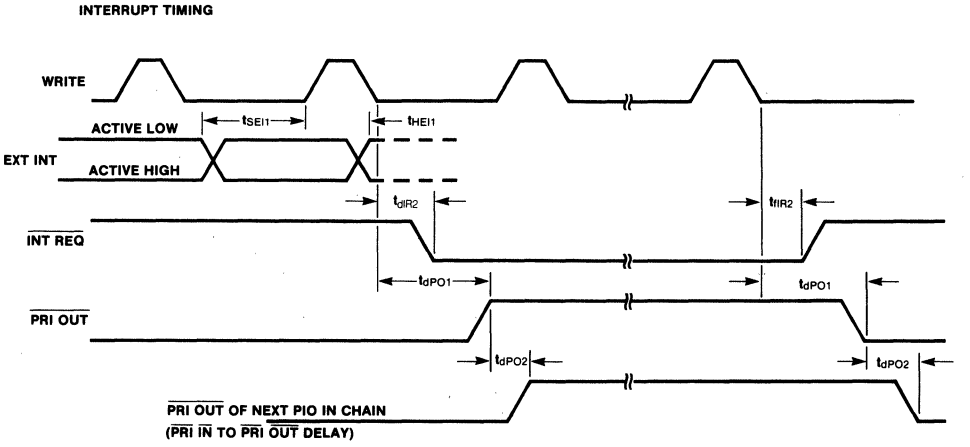


Table 5 F3871 Clock Timing Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
$P\phi$	Clock Period	0.5		10	μs	
P_0	Low time	180			ns	
P_1	High time	180			ns	
PW	WRITE Clock Period		$4P\phi$			Short cycle
PW_0	WRITE Clock Period		$6P\phi$			Long cycle
PW_1	WRITE Pulse Width	$P\phi-100$		$P\phi$		
t_{dw1}	ϕ – to WRITE + delay			250	ns	
t_{dw0}	ϕ – to WRITE – delay			225	ns	

Table 6 F3871 Strobe Timing Characteristics

Symbol	Parameter	Min.	Max.	Units	Comments
$t_{I/O-S}$	Port Output to $\overline{\text{STROBE}}$ delay	$3t\phi-1000$	$3t\phi+250$	ns	Note 1
t_{SL}	$\overline{\text{STROBE}}$ Pulse Width, Low	$8t\phi-250$	$12t\phi+250$	ns	
$t_{w-I/O}$	WRITE to I/O Port Output Valid		1000	ns	Note 2

NOTES
1. Load is 50 pF plus 3 standard TTL inputs.
2. Load is 50 pF plus 1 standard TTL input.

Table 7 F3871 Input Timing Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
t_{SR2}	ROMC Valid Measured from Fall of WRITE			550	ns	
t_{HR1}	ROMC Required Hold After Fall of WRITE	20			ns	
t_{SD4}	Data Bus Setup Time				ns	
t_{HD3}	Data Input Hold Time	20			ns	
$t_{SI/02}$	I/O Input Setup Time	1.3			ns	
$t_{HI/02}$	I/O Input Hold Time	20			ns	

Table 8 F3871 Output Timing Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
t_{fDR1}	WRITE to $\overline{DBDR}$ Floating			400	ns	
t_{dDR1}	ϕ to $\overline{DBDR}$ 1-0		200	625	ns	$C_L = 100$ pF; $R_L = 12.5$ kΩ
t_{dDR2}	WRITE to $\overline{DBDR}$ 1-0			$2P\phi + 625 - t_{dw0}$	ns ns	$C_L = 100$ pF; $R_L = 12.5$ kΩ $C_L = 100$ pF
t_{dD3}	WRITE to Data Valid	$2P\phi - t_{dW0}$	$2P\phi - 400$	$2P\phi + 700 - t_{dw0}$	ns	$C_L = 100$ pF
t_{0HD2}	Guaranteed Data Hold Time After Fall of WRITE	30			ns	

Table 9 F3871 Interrupt Timing Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
t_{SEI1}	EXT INT Setup Time	750			ns	
t_{HEI}	EXT INT Hold Time	30			ns	
t_{dIR2}	WRITE to $\overline{INT\ REQ}$ Delay			430	ns	$C_L = 100$ pF
t_{PO1}	WRITE to $\overline{PRI\ OUT}$ Delay			640	ns	$C_L = 50$ pF
t_{dPO2}	$\overline{PRI\ IN}$ to $\overline{PRI\ OUT}$ Delay			350	ns	$C_L = 50$ pF
t_{fIR1}	WRITE to $\overline{INT\ REQ}$ Float by PIO			640	ns	Open Drain Output

DC Characteristics

The DC characteristics of the F3871 PIO are supplied in table 10.

Table 10 F3871 DC Characteristics

Symbol	Parameter	Signal	Min.	Max.	Units	Test Conditions
V_{IH}	Input High Voltage	Data Bus (DB_0 -DB_7)	2.0	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
V_{OH}	Output High Voltage		3.9	V_{DD}	V	$I_{OH} = -100\ \mu A$
V_{OH}	Output High Voltage		2.4		V	$I_{OH} = 100\ \mu A$, $V_{GG} = 5\ V \pm 5\%$
V_{OL}	Output Low Voltage			0.4	V	$I_{OL} = 1.6$ mA
I_{IH}	Input High Current			1.0	μA	$V_{IN} = 6$ V, 3-state mode
I_{OL}	Input Low Current			−1.0	μA	$V_{IN} = V_{SS}$, 3-state mode
C_I	Input Capacitance			10	pF	3-state mode

Table 10 F3871 DC Characteristics

Symbol	Parameter	Signal	Min.	Max.	Units	Test Conditions
V_{IH}	Input High Voltage	Clock Lines (ϕ, WRITE)	2.0	V_{DD}	V	
V_{IL}	Input Low Voltage		V_{CC}	0.8	V	
I_L	Leakage Current			±1.0	μA	$V_{IN} = V_{SS}$ to +6 V
C_I	Input Capacitance			10	pF	
V_{IH}	Input High Voltage	Priority In and Control	3.5	V_{DD}	V	
V_{IL}	Input Low Voltage	Lines ($\overline{PRI\ IN}$, $ROMC_0$ -	V_{CC}	0.8	V	
I_L	Leakage Current	$ROMC_4$)		1.0	μA	$V_{IN} = V_{SS}$ to +6 V
C_I	Input Capacitance			10	pF	
V_{OH}	Output High Voltage	Priority out ($\overline{PRI\ OUT}$)	3.9	V_{DD}	V	$I_{OH} = -100$ μA
V_{OL}	Output Low Voltage		V_{ss}	0.4	V	$I_{OL} = -1.8$ mA
V_{OH}	Output High Voltage	Interrupt Request			V	Open Drain Output[1]
V_{OL}	Output Low Voltage	($\overline{INT\ REQ}$)	V_{SS}	0.4	V	$I_{OL} = 0.8$ mA
I_L	Leakage Current			1.0	μA	$V_{IN} = 6$ V, Output Device Off
C_I	Input Capacitance			10	pF	Output Device Off
V_{OH}	Output High Voltage	Data Bus Drive ($\overline{DBDR}$)				External Pull-up
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 1.8$ mA
I_L	Leakage Current			1.0	μA	$V_{IN} = 6$ V, Output Device Off
C_I	Input Capacitance			10	pF	Output Device Off
V_{IH}	Input High Voltage	External Interrupt	2.0		V	External Pull-ups Exist
V_{IL}	Input Low Voltage	($\overline{EXT\ INT}$)		0.8	V	Internal
I_{IH}	Input High Current			−1.6	μA	$V_{IN} = 0.4$ V
I_{IL}	Input Low Current		100		μA	$V_{IN} = 2.4$ V
C_I	Input Capacitance			10	pF	
V_{OH}	Output High Voltage	Strobe ($\overline{STROBE}$)	2.4	V_{DD}	V	$I_{OH} = -300$ μA
V_{OL}	Output Low Voltage		V_{SS}	0.4	V	$I_{OL} = 5.0$ mA
V_{OH}	Output High Voltage	I/O Port Option A	2.4	V_{DD}	V	$I_{OH} = -100$ μA
V_{OL}	Output Low Voltage	(Standard Pull-up)	V_{SS}	0.4	V	$I_{OL} = 1.8$ mA
V_{IH}	Input High Voltage		2.0	V_{DD}	V	Internal Pull-up to V_{DD}
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_{IL}	Input Low Current			1.0	mA	$V_{IN} = 0.4$ V[2]
C_I	Input Capacitance			10	pF	
V_{OH}	Output High Voltage	I/O Port Option B				External Pull-up
V_{OL}	Output Low Voltage	(Open Drain)	V_{SS}	0.4	V	$I_{OL} = 1.8$ mA
V_{IH}	Input High Voltage		2.0	6.0	V	
V_{IL}	Input Low Voltage		V_{SS}	0.8	V	
I_L	Leakage Current			1.0	μA	$V_{IN} = 6$ V, Output Device Off
C_I	Input Capacitance			10	pF	
V_{OH}	Output High Voltage	I/O Port Option C	1.5	V_{DD}	V	$I_{OH} = -1.5$ mA
V_{OL}	Output Low Voltage	(Driver Pull-up)	V_{SS}	0.4	V	$I_{OL} = 1.8$ mA
I_{OH}	Output High Current		−1.5	−9.0	mA	$V_{OH} = 0.7$ V to 1.5 V

NOTES
1. Pull-up resistor to V_{DD} on CPU.
2. Measured with a high-level I/O port output.
3. Positive current is defined as conventional current flowing into the pin referenced.
4. $V_{SS} = 0$ V, $V_{DD} = +5$ V ±5%, $V_{GG} = +12$ V ±5%, $T_A = 0°C$ to +70°C unless otherwise noted.

The supply currents are given in table 11.

Table 11 Supply Currents

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
I_{DD}	V_{DD} Current		30	70	mA	f = 2 MHz, Outputs Unloaded
I_{GG}	V_{GG} Current		10	18	mA	f = 2 MHz, Outputs Unloaded

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

V_{GG}	+15 V, −0.3 V
V_{DD}	+7 V, −0.3 V
External Interrupt Input	−600 μA, +225 μA
All Other Inputs and Outputs	+7 V, −0.3 V
Storage Temperature	−55 °C, +150 °C
Operating Temperature	0 °C, +70 °C

NOTE: All voltages are with respect to V_{SS}.

Recommended Operating Ranges

The recommended operating ranges of the PIO devices are shown below.

Symbol	Parameter	Min.	Typ.	Max.
V_{DD}	Supply Voltage	+4.75 V	+5 V	+5.25 V
V_{GG}		+11.4 V	+12 V	+12.6 V
V_{SS}			0 V	

Ordering Information

Part Number	Package	Temperature Range
F3871DC	Ceramic	C
F3871DM	Ceramic	M
*F3871PC	Plastic	C

C = Commercial Temperature Range 0° to +70°C
L = Limited Temperature Range −40°C to +85°C
M = Military Temperature Range −55°C to +125°C
*Versions E, F, G, and H are stocked items.

FAIRCHILD
A Schlumberger Company

Section 4
Controller Family

F387X Family

The Fairchild F387X family of devices represents a line of complete, 8-bit microcomputers on single MOS integrated circuits (see figure 4-1). Fabricated using Fairchild double-ion-implanted, N-channel silicon-gate technology and advanced circuit design techniques, the F387X family offers maximum cost-effectiveness in a variety of logic replacement and control applications.

Figure 4-1 F387X Family Organization

F3870 SINGLE-CHIP MICROCOMPUTER
F3870A/F3870B HIGH-SPEED SINGLE-CHIP MICROCOMPUTER
F38C70 SINGLE-CHIP MICROCOMPUTER
F38E70 SINGLE-CHIP MICROCOMPUTER
F3872/F38L72 SINGLE-CHIP MICROCOMPUTER

All F387X family microcomputers execute the F8 instruction set of more than 70 commands. They are available with a wide range of memory types and sizes, allowing the designer to select the best combination of RAM and ROM for a particular application. The F387X devices are also available with special types of I/O.

The F387X family devices are all pin-compatible, permitting easy system upgrading by replacement of one device in an application with another family member having greater quantities of RAM or ROM, special I/O functions, or all three. Because of this simple upgrading, an F387X-based microcomputer can be enhanced or expanded in many different ways without affecting system printed circuit board, enclosure, or power supply requirements.

The Fairchild F387X single-chip microcomputer family is recognized as an industry standard in logic replacement and control. The devices have been designed into, and successfully used in, a wide range of applications requiring intelligent control.

Part Numbers

Because of the on-going growth of the F387X family, a new numbering system for these devices has been developed by Fairchild:

F38 7X

Technology/Option
- Omit = NMOS
- C = CMOS
- E = EPROM
- L = Low-power stand-by option (F3872 only)

Speed Grade
- Omit = Standard
- A = 1.33 us cycle time
- B = 1 us cycle time

ROM size
- 1 = 1024 bytes
- 2 = 2048 bytes
- 3 = 3072 bytes
- 4 = 4032 or 4096 bytes

Descriptions

Following is data that describes the members of the F387X single-chip microcomputer family.

FAIRCHILD
A Schlumberger Company

F3870
Single-Chip Microcomputer

Microprocessor Product

Description

The Fairchild single-chip microcomputer series offers a variety of circuits to serve the high-volume, cost-sensitive controller market. The F3870 is a complete 8-bit microcomputer on a single MOS integrated circuit. The F3870 can execute the F8 instruction set of more than 70 commands, allowing expansion into multi-chip configurations with software compatibility. The device features 64 bytes of scratchpad RAM, a programmable binary timer, 32 bits of I/O, a single +5 V power supply requirement, and a choice of 1K, 2K, 3K, or 4K of ROM.

Utilizing Fairchild's double-ion-implanted, N-channel silicon-gate technology and advanced circuit design techniques, the single-chip F3870 offers maximum cost effectiveness in a wide range of control and logic replacement applications.

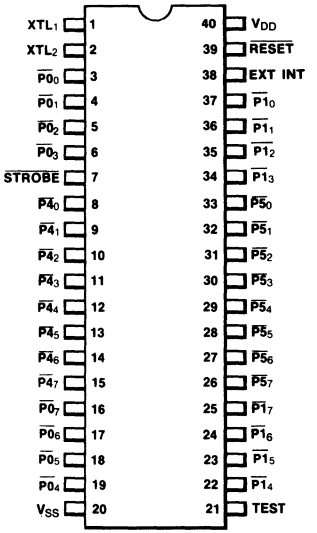
- **Single-Chip Microcomputer**
- **Software-Compatible with F8 Family**
- **1024-, 2048-, 3072-, or 4096-Byte Programmable ROM**
- **64-Byte Scratchpad RAM**
- **32-Bit (4-Port) TTL-Compatible I/O**
- **Programmable Binary Timer:**
  - **Interval Timer Mode**
  - **Pulse Width Measurement Mode**
  - **Event Counter Mode**
- **External Interrupt**
- **Crystal, LC, RC, or External Time Base**
- **Low Power (275 mW, Typical)**
- **Single +5 V ± 10% Power Supply**

Signal Functions

The functions of the F3870 inputs and outputs are described in *Table 1.*

Connection Diagram

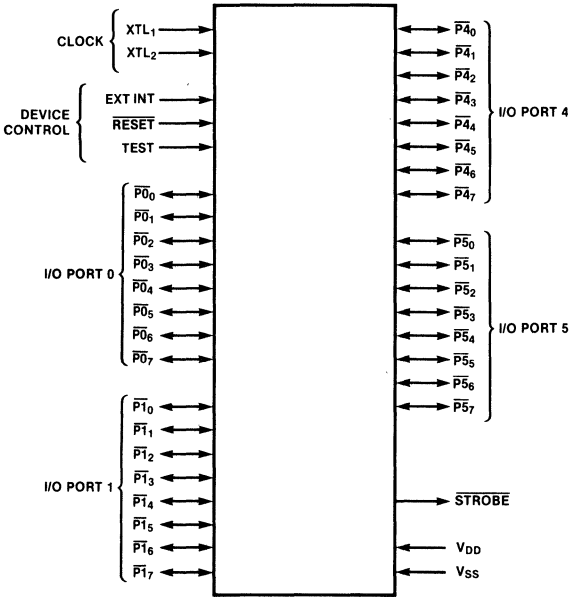
40-Pin DIP



(Top View)

4

Signal Functions



Device Organization

This section describes the basic functional elements of the F3870 shown in *Figures 1* and *2.*

Main Control Logic

The instruction register (IR) receives the operation code (op code) of the instruction to be executed from the program ROM via the data bus. During all op code fetches, eight bits are latched into the IR. Some instructions are completely specified by the upper four bits of the op code; in such instructions, the lower four bits are an immediate register address or an immediate 4-bit operand. Once latched into the IR, the main control logic decodes the instruction and provides the necessary control gating signals to all circuit elements.

ROM Address Registers

There are four 12-bit registers associated with the program ROM of the F3870. *(In the F3870-1, -2, and -3, the 12-bit registers can address more memory space than is physically available on the chip; user caution is advised. Older versions of the F3870-1 and -2, predating date code 8213, may have 11-bit registers; contact Fairchild if you have any questions.)* These are the program counter (P0), the stack register (P), the data counter (DC), and the auxiliary data counter (DC1). The program counter is used to address instructions or immediate operands. The stack register is used to save the contents of P0 during an interrupt or subroutine call. Thus, P contains the return address at which processing is to resume upon completion of the subroutine or the interrupt routine.

The data counter is used to address data tables. This register is autoincrementing. Of the two data counters, only DC can access the ROM. However, the XDC instruction allows DC and DC1 to be exchanged.

Associated with the F3870 address registers is a 12-bit adder/incrementer. This logic element is used to increment P0 or DC when required and is also used to add displacements to P0 on relative branches or to add the data bus contents to DC in the add data counter (ADC) instruction.

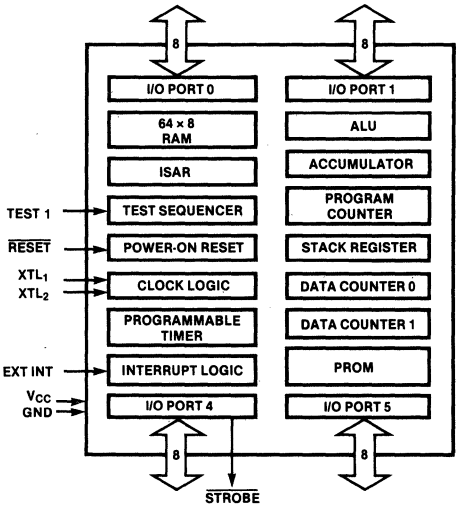
Program ROM

The microcomputer program and data constants are stored in the program ROM, which may be 1024 × 8 (F3870-1), 2048 × 8 (F3870-2), 3072 × 8 (F3870-3), or 4096 × 8 (F3870-4) bytes. When a ROM access is required, the appropriate address register (P0 or DC) is gated onto the ROM address bus and the ROM output is gated onto the main data bus. The first byte in the ROM is location zero.

Table 1 Signal Functions

Mnemonic	Pin No.	Name	Description
Device Control			
EXT INT	38	External Interrupt	Software-programmable input that is also used in conjunction with the timer for pulse width measurement and event counting.
$\overline{\text{RESET}}$	39	External Reset	Input that may be used to externally reset the F3870. When pulled low, the F3870 resets; when then allowed to go high, the F3870 begins program execution at program location H'0000'.
TEST	21	Test Line	An input used only in testing the F3870. For normal circuit operation, TEST is left unconnected or grounded.
Clock			
$\overline{\text{STROBE}}$	7	Ready Strobe	Normally high output that provides a single low pulse after valid data is present on the $\overline{P4_0}$-$\overline{P4_7}$ pins during an output instruction.
XTL_1, XTL_2	1, 2	Time Base	Inputs to which a crystal (1 MHz to 4 MHz), LC network, RC network, or external single-phase clock may be connected.
I/O Ports			
$\overline{P0_0}$-$\overline{P0_7}$ $\overline{P1_0}$-$\overline{P1_7}$ $\overline{P4_0}$-$\overline{P4_7}$ $\overline{P5_5}$-$\overline{P5_7}$	3-6, 8-19, 22-37	I/O Ports	Thirty-two bidirectional lines that can be individually used as either TTL-compatible inputs or latched outputs.
Power			
V_{DD}	40	Power Input	+5 V ± 10% power supply
V_{SS}	20	Ground	Signal and power ground

Fig. 1 F3870 Architecture

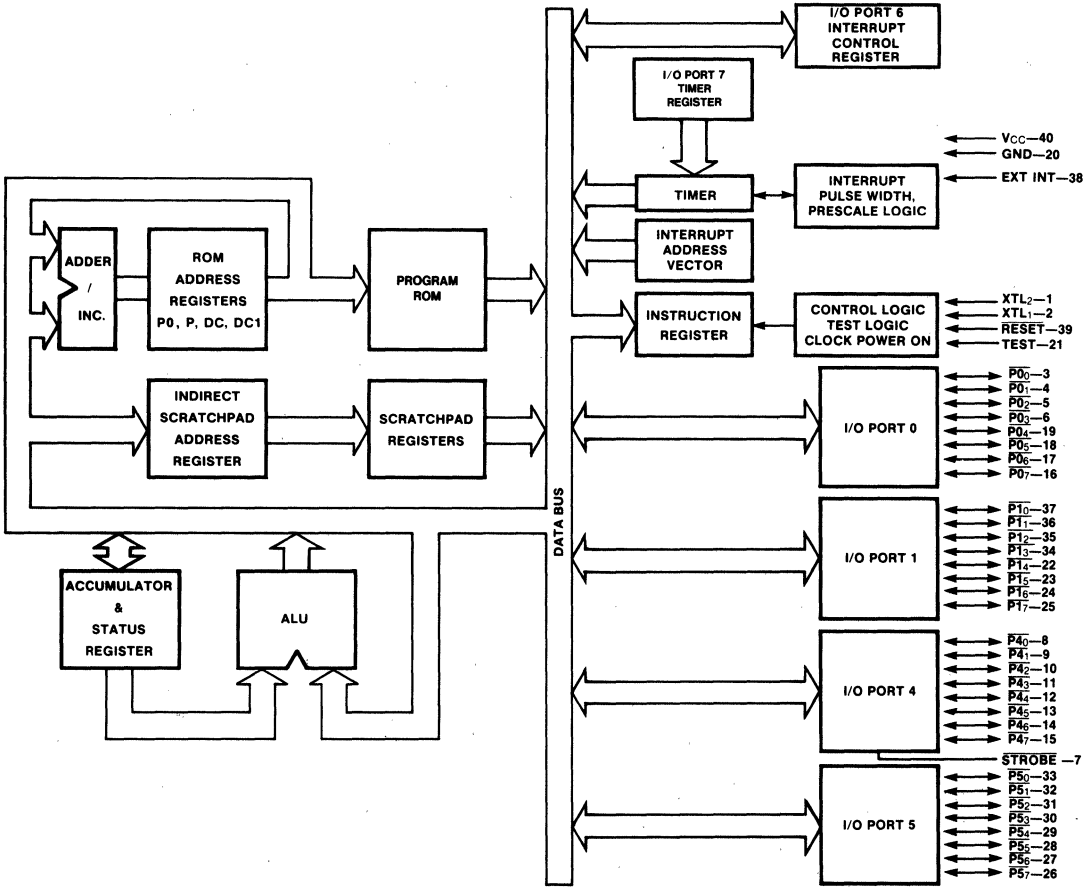


Scratchpad and ISAR

The scratchpad provides 64 8-bit registers that may be used as general-purpose RAM. The indirect scratchpad address register (ISAR) is a 6-bit register used to address the 64 registers. All 64 registers may be accessed using the ISAR. In addition, the lower order 12 registers may also be directly addressed.

The ISAR can be visualized as holding two octal digits. This division of the ISAR is important, since a number of instructions increment or decrement only the least significant three bits of the ISAR when referencing scratchpad bytes via the ISAR. This makes it easy to reference a buffer consisting of contiguous scratchpad bytes. For example, when the low-order octal digit is incremented or decremented, the ISAR is incremented from octal 27 to 20 or is decremented from octal 20 to 27. This feature of the ISAR is very useful in many program sequences. All six bits of the ISAR may be loaded at one time, or either half may be loaded independently.

Fig. 2 F3870 Block Diagram



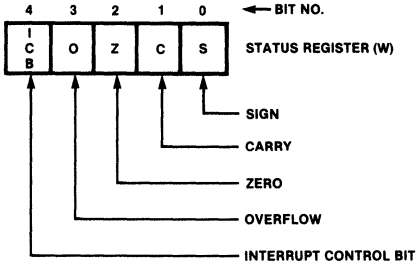
Scratchpad registers 9 through 15 (decimal) are given mnemonic names (J, H, K, and Q) because of special linkages between these registers and other registers, such as the stack register. These special linkages facilitate the implementation of multi-level interrupts and subroutine nesting. For example, the instruction LR K, P stores the lower eight bits of the stack register in register 13 (K lower, or KL) and stores the upper four bits of P in register 12 (K upper, or KU).

Arithmetic and Logic Unit (ALU)

After receiving commands from the main control logic, the ALU performs the required arithmetic or logic operations (using the data presented on the two input buses) and provides the result on the result bus. The arithmetic operations that can be performed in the ALU are binary add, decimal adjust, add with carry, decrement, and increment. The logic operations that can be performed are AND, OR, exclusive-OR, ones complement, shift right, and shift left. Besides providing the result on the result bus, the ALU also provides four signals presenting the status of the result. These signals, stored in the status register (W), represent the CARRY, OVERFLOW, SIGN, and ZERO conditions of the result of the operation.

Accumulator
The accumulator (ACC) is the principal register for data manipulation within the F3870. The ACC serves as one input to the ALU for arithmetic or logical operation. The results of ALU operations are stored in the ACC.

Status Register
The status register (also referred to as the W register) holds five status flags, as follows:



Summary of Status Bit

$$\text{OVERFLOW} = \text{CARRY}_7 \oplus \text{CARRY}_6$$

$$\text{ZERO} = \overline{ALU_7} \wedge \overline{ALU_6} \wedge \overline{ALU_5} \wedge \overline{ALU_4} \wedge \overline{ALU_3} \wedge \overline{ALU_2} \wedge \overline{ALU_1} \wedge \overline{ALU_0}$$

$$\text{CARRY} = \text{CARRY}_7$$

$$\text{SIGN} = \overline{ALU_7}$$

The interrupt control bit (ICB) of the status register may be used to allow or disallow interrupts in the F3870. This bit is not the same as the two interrupt enable bits in the interrupt control port (ICP). If the ICB is set and the F3870 interrupt logic communicates an interrupt request to the CPU section, the interrupt is acknowledged and processed upon completion of the first non-privileged instruction. If the ICB is cleared, an interrupt request is not acknowledged or processed until the ICB is set.

I/O Ports
The F3870 provides four complete bidirectional I/O ports; these are ports 0, 1, 4, and 5. In addition, the interrupt control register is addressed as port 6 and the binary timer is addressed as port 7. An output instruction (OUT or OUTS) causes the contents of the ACC to be latched into the addressed port. An input instruction (IN or INS) transfers the contents of the port to the ACC (port 6 is an exception that is described later). The I/O pins on the F3870 are logically inverted. The schematic of an I/O pin and conceptual illustrations of available output drive options are shown in *Figure 3.*

An output ready strobe is associated with port 4. This flag may be used to signal a peripheral device that the F3870 has just completed a single low pulse shortly after the output operation is completely finished, so either edge may be used to signal the peripheral. This STROBE signal may also be used to request new input information from a peripheral simply by doing a dummy output of H'00' to port 4 after completing the input operation.

Timer and Interrupt Control Port
The timer is an 8-bit binary down counter that is software-programmable to operate in one of three modes: the interval timer mode, the pulse width measurement mode, or the event counter mode; the timer characteristics are described in *Table 2.* As shown in *Figure 4,* associated with the timer is an 8-bit register called the interrupt control port, a programmable prescaler, and an 8-bit modulo-N register; a functional logic diagram is shown in *Figure 5.*

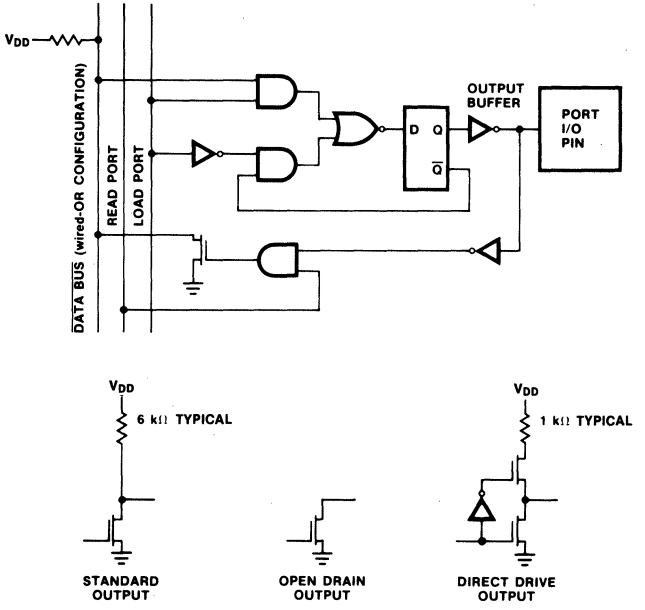
The desired timer mode, prescale value, starting and stopping the timer, active level of the EXT INT pin, and local enabling or disabling of interrupts are selected by outputting the proper bit configuration from the accumulator to the ICP (port 6) with an OUT or OUTS instruction. Bits within the ICP are defined as follows:

Interrupt Control Port (Port 6)

Bit 0—External Interrupt Enable
Bit 1—Timer Interrupt Enable
Bit 2—EXT INT Active Level
Bit 3—Start/Stop Timer
Bit 4—Pulse Width/Interval Timer
Bit 5— ÷ 2 Timer Prescale Values
Bit 6— ÷ 5 Timer Prescale Values
Bit 7— ÷ 20 Timer Prescale Values

A special situation exists when reading the ICP with an IN or INS instruction. The accumulator is not loaded with the contents of the ICP; instead, accumulator bits 0 through 6 are loaded with zeros, while bit 7 is loaded with the logic level being applied to the EXT INT pin, thus allowing the status of the EXT INT pin to be determined without the necessity of servicing an external interrupt request. This capability is useful in establishing a high-speed, polled handshake procedure or for using EXT INT as an extra input pin if external interrupts are not required and the timer is used only in the interval timer mode.

Fig. 3 I/O Port Diagram



Ports 0 and 1 are standard output type only.
Ports 4 and 5 may be any of the three output options, each pin individually assignable to any port.
The $\overline{\text{STROBE}}$ output is always configured similar to a standard output, except that it is capable of driving three TTL loads.

Fig. 4 Timer and Interrupt Control Port Block Diagram

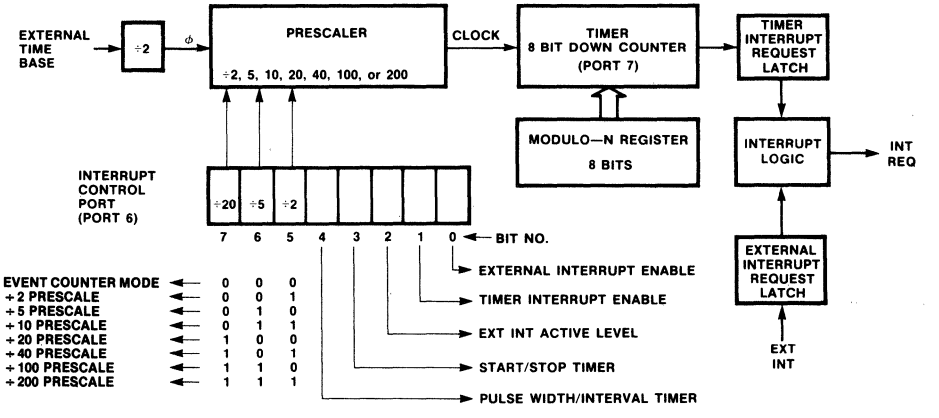


Fig. 5 Timer/Interrupt Functional Diagram

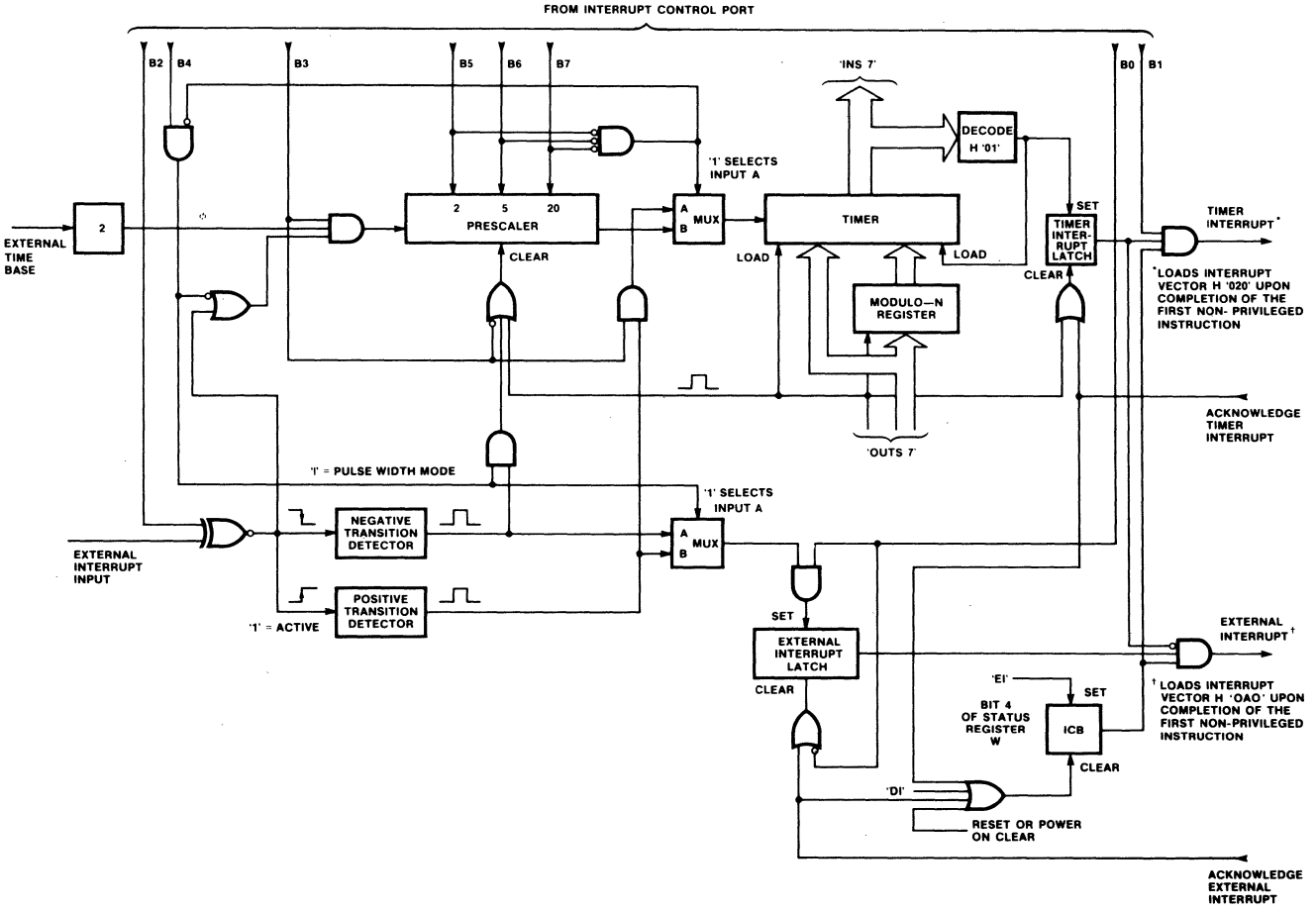


Table 2 Timer Characteristics

Characteristic	Value
Interval Timer Mode	
Single Interval Error, Free-Running (Note 3)	$\pm 6t\phi$
Cumulative Interval Error, Free-Running (Note 3)	0
Error Between Two Timer Reads (Note 2)	$\pm (tpsc + t\phi)$
Start Timer to Stop Timer Error (Notes 1, 4)	$+t\phi$ to $-(tpsc + t\phi)$
Start Timer to Read Timer Error (Notes 1, 2)	$-5t\phi$ to $-(tpsc + 7t\phi)$
Start Timer to Interrupt Request Error (Notes 1, 3)	$-2t\phi$ to $-8t\phi$
Load Timer to Stop Timer Error (Note 1)	$+t\phi$ to $-(tpsc + 2t\phi)$
Load Timer to Read Timer Error (Notes 1, 2)	$-5t\phi$ to $-(tpsc + 8t\phi)$
Load Timer to Interrupt Request Error (Notes 1, 3)	$-2t\phi$ to $-9t\phi$
Pulse Width Measurement Mode	
Measurement Accuracy (Note 4)	$+t\phi$ to $-(tpsc + 2t\phi)$
Minimum Pulse Width of EXT INT Pin	$2t\phi$
Event Counter Mode	
Minimum Active Time of EXT INT Pin	$2t\phi$
Minimum Inactive Time of EXT INT Pin	$2t\phi$

Definitions
Error = indicated time value – actual time value
tpsc = $t\phi$ × prescale value

Notes
1. All times that entail loading, starting, or stopping the timer are referenced from the end of the last machine cycle of the OUT or OUTS instruction.
2. All times that entail reading the timer are referenced from the end of the last machine cycle of the IN or INS instruction.
3. All times that entail the generation of an interrupt request are referenced from the start of the machine cycle in which the appropriate interrupt request latch is set. Additional time may elapse if the interrupt request occurs during a privileged or multi-cycle instruction.
4. Error may be cumulative if operation is repetitively performed.

The rate at which the timer is clocked in the interval timer mode is determined by the frequency of an internal ϕ clock and by the division value selected for the prescaler. (The internal clock operates at one-half the external time base frequency.) If ICP bit 5 is set and bits 6 and 7 are cleared, the prescaler divides ϕ by 2. Likewise, if bit 6 or 7 is individually set, the prescaler divides ϕ by 5 or 20, respectively. Combinations of bits 5, 6, and 7 may also be selected. For example, if bits 5 and 7 are set while bit 6 is cleared, the prescaler divides by 40. Thus, possible prescaler values are: ÷ 2, ÷ 5, ÷ 10, ÷ 20, ÷ 40, ÷ 100, and ÷ 200.

Any of three conditions cause the prescaler to be reset: whenever the timer is stopped by clearing ICP bit 3, on execution of an output instruction to port 7 (the timer is assigned port address 7), or on the trailing edge transition of the EXT INT pin when in the pulse width measurement mode. These last two conditions are explained in the following paragraphs.

An OUT or OUTS instruction to port 7 loads the contents of the accumulator into both the timer and the 8-bit modulo-N register, resets the prescaler, and clears any previously stored timer interrupt request. As previously noted, the timer is an 8-bit down counter that is clocked by the prescaler in the interval timer mode and in the pulse width measurement mode. The prescaler is not used in the event counter mode. The modulo-N register is a buffer whose function is to save the value that was most recently output to port 7. The modulo-N register is used in all three timer modes.

Interval Timer Mode — When ICP bit 4 is cleared (logic 0) and at least one prescale bit is set, the timer operates in the interval timer mode. When bit 3 of the ICP is set, the timer starts counting down from the modulo-N value. After counting down to H'01', the timer returns to the

modulo-N value at the next count. On the transition from H'01' to H'N', the timer sets a timer interrupt request latch. Note that the interrupt request latch is set by the transition of H'N' in the timer, thus allowing a full 256 counts if the modulo-N register is preset to H'00'. If bit 1 of the ICP is set, the interrupt request is passed to the CPU section of the F3870. However, if bit 1 of the ICP is a logic 0, the interrupt request is not passed, but the interrupt request latch remains set. If ICP bit 1 is subsequently set, the interrupt request is then passed to the CPU. Only two events can reset the timer interrupt request latch: when the timer interrupt request is acknowledged by the CPU, or when a new load of the modulo-N register is performed.

Consider an example in which the modulo-N register is loaded with H'64' (decimal 100). The timer interrupt request latch is set at the 100th count following the timer start, and the timer interrupt request latch is repeatedly set on precise 100-count intervals. If the prescaler is set at ÷ 40, the timer interrupt request latch is set every 4000 ϕ clock periods. For a 2 MHz ϕ clock (4 MHz time base frequency), this produces 2 ms intervals.

The range of possible intervals is from 2 to 51,200 ϕ clock periods (1 μs to 25.6 ms for a 2 MHz clock). However, approximately 50 ϕ periods is a practical minimum because the time between setting the interrupt request latch and the execution of the first instruction of the interrupt service routine is at least 29 ϕ periods (the response time is dependent upon how many privileged instructions are encountered when the request occurs). To establish time intervals greater than 51,200 clock periods is simply a matter of using the timer interrupt service routine to count the number of interrupts, saving the result in one or more of the scratchpad registers until the desired interval is achieved. With this technique, virtually any time interval, or several time intervals, may be generated.

The timer may be read at any time and in any mode using an input instruction (IN 7 or INS 7); this may take place on-the-fly without interfering with normal timer operation. The timer may also be stopped at any time by clearing bit 3 of the ICP. The timer holds its current contents indefinitely and resumes counting when bit 3 is again set. The prescaler, however, is reset whenever the timer is stopped; thus, a series of starts and stops results in a cumulative truncation error.

For a free-running timer in the interval timer mode, the time interval between any two interrupt requests may be in error by ± 6 ϕ clock periods, although the cumulative error over many intervals is zero. The prescaler and timer generate precise intervals for setting the timer interrupt request latch, but the time-out may occur at any time within a machine cycle. (There are two types of machine cycles: short cycles that consist of four ϕ clock periods, and long cycles that consist of 6 ϕ clock periods.) In the multi-chip F8 family, there is a signal referred to as the write clock, which corresponds to a machine cycle. Interrupt requests are synchronized with the internal write clock, thus giving rise to the possible ± 6 ϕ error. Additional errors may arise due to the interrupt request occurring while a privileged instruction or multi-cycle instruction is being executed. Nevertheless, for most applications, all of the above errors are negligible, especially if the desired time interval is greater than 1 ms.

Pulse Width Measurement Mode — When ICP bit 4 is set (logic 1) and at least one prescale bit is set, the timer operates in the pulse width measurement mode. This mode is used for accurately measuring the duration of a pulse applied to the EXT INT pin. The timer is stopped and the prescaler is reset when the EXT INT pin is at its inactive level. The active level of EXT INT is defined by ICP bit 2: if cleared, EXT INT is active-low; if set, EXT INT is active-high. If ICP bit 3 is set, the prescaler and timer start counting when EXT INT transfers to the active level. When EXT INT returns to the inactive level, the timer stops, the prescaler resets, and, if ICP bit 0 is set, an external interrupt request latch is set. (Unlike timer interrupts, external interrupts are not latched if the ICP interrupt enable bit is not set.)

As in the interval timer mode, the timer may be read at any time, or may be stopped at any time by clearing ICP bit 3, the prescaler and the ICP bit 1 function as previously described; the timer still functions as an 8-bit binary down counter with the timer interrupt request latch being set on the timer's transition from H'01' to H'N' (modulo-N value). Note that the EXT INT pin has nothing to do with loading the timer; its action is that of automatically starting and stopping the timer and of generating external interrupts. Pulse widths longer than the prescaler value times the modulo-N value are easily measured by using the timer interrupt service routine to store the number of timer interrupts in one or more scratchpad registers.

As for accuracy, the actual pulse duration is typically slightly longer than the measured value because the status of the prescaler is not readable and is reset when the timer is stopped. Thus, for maximum accuracy, it is advisable to use a small-division setting for the prescaler.

Event Counter Mode — When ICP bit 4 is cleared and all prescale bits (ICP bits 5, 6, and 7) are cleared, the timer operates in the event counter mode. This mode is used for counting pulses applied to the EXT INT pin. If ICP bit 3 is set, the timer decrements on each transition from the inactive level to the active level of the EXT INT pin. The prescaler is not used in this mode but, as in the other two timer modes, the timer may be read at any time, or may be stopped at any time by clearing ICP bit 3; ICP bit 1 functions are previously described, and the timer interrupt request latch is set on the timer's transition from H'01' to H'N' (modulo-N value).

Normally, ICP bit 0 should be kept cleared in the event counter mode; otherwise, external interrupts are generated on the transition from the inactive level to the active level of the EXT INT pin.

For the event counter mode, the minimum pulse width required on the EXT INT pin is 2 ϕ clock periods, and the minimum inactive time is 2 ϕ clock periods; therefore, the maximum repetition rate is 500 Hz.

External Interrupts

When the timer is in the interval timer mode, the EXT INT pin is available for non-timer-related interrupts. If ICP bit 0 is set, an external interrupt request latch is set when there is a transition from the inactive level to the active level of the EXT INT pin (EXT INT is an edge-triggered input). The interrupt request is latched until either acknowledged by the CPU or ICP bit 0 is cleared (unlike timer interrupt requests, which remain latched even when ICP bit 1 is cleared). External interrupts are handled in the same fashion when the timer is in the pulse width measurement mode or in the event counter mode, except that in the pulse width measurement mode the external interrupt request latch is set on the trailing edge of the EXT INT input; that is, on the transition from the active level to the inactive level.

Interrupt Handling

When either a timer or an external interrupt request is communicated to the CPU section of the F3870, it is acknowledged and processed at the completion of the first non-privileged instruction if the interrupt control bit of the status register is set. If the interrupt control bit is not set, the interrupt request continues either until the interrupt control bit is set and the CPU acknowledges the interrupt or until the interrupt request is cleared as previously described.

If there are a timer interrupt request and an external interrupt request when the CPU starts to process the requests, the timer interrupt is handled first.

When an interrupt is allowed, the CPU requests that the interrupting element pass its interrupt vector address to the program counter via the data bus. The vector address for a timer interrupt is H'20'; the vector address for an external interrupt is H'0A0'. After the vector address is passed to the program counter, the CPU sends an acknowledge signal to the appropriate interrupt request latch, which clears that latch. The execution of the interrupt service routine then commences. The return address of the original program is automatically saved in the stack register, P.

Power-On Clear

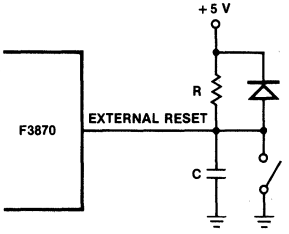
The F3870 contains power-on clear circuitry to automatically reset the internal logic following the application of external power. Since many variations of power supply circuitry exist, Fairchild cannot guarantee that the power-on clear will operate under every power-up condition.

The power-on clear circuitry contains on-chip sensors to monitor various conditions. The following conditions must be satisfied before the power-reset sequence is allowed to start:

1. Supply voltage must be above a certain value, typically +3 V to +4 V.
2. The clocks of the device must be functioning.
3. The substrate bias must reach a certain level.

All three conditions must be met before the power-on clear circuitry initiates a reset cycle. However, these conditions can be satisfied even with a supply voltage of as low as 3 volts. The latest versions of the F3870 have a modified delay circuit that gives a typical delay of 500 μs (with a 4 MHz crystal) after the above conditions are met. This is an improvement over the earlier F3870 versions.

Since the F3870 is only guaranteed to operate at a supply voltage of 4.5 V or greater, the user must ensure that the supply voltage is at least 4.5 V when the F3870 initiates the reset cycle. For power supplies having a slow rise time, an external RC network can be converted to the external reset input of the F3870 to hold the device in a reset state long enough to allow the power supply to reach a voltage of 4.5 V.



External Reset
When the $\overline{\text{RESET}}$ input is low, the contents of the program counter are pushed to the stack register and the program counter and the ICB of the status register are cleared. The original stack register contents are lost. As with power-on clear, ports 4, 5, 6, and 7 are loaded with H'00'. The contents of all other registers and ports are unchanged. When $\overline{\text{RESET}}$ is high, the first program instruction is fetched from ROM location H'0000'.

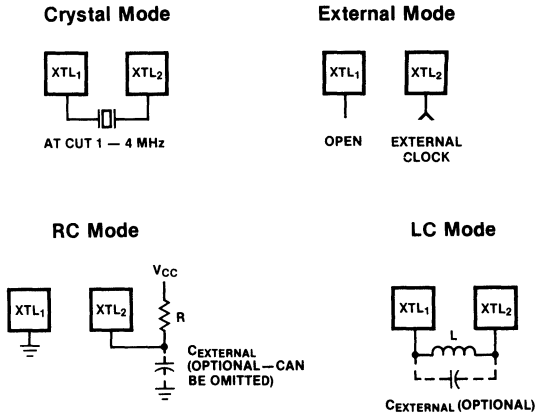
Test Logic
Special test logic is implemented to allow access to the internal main data bus for test purposes.

In normal operation, the TEST pin is unconnected or is connected to ground. When TEST is placed at a level of from 2.8 V to 3.0 V, port 4 becomes an output of the internal data bus and port 5 becomes a wired-OR input to the internal data bus. The data appearing on the port 4 pins is logically true, whereas input data forced on port 5 must be logically false. When TEST is placed at a high level (8.8 V to 9.0 V), the ports act as described above and, additionally, the program ROM is prevented from driving the data bus. In this mode, operands and instructions may be forced externally through port 5 instead of being accessed from the program ROM. When TEST is in either the TTL state or the high state, $\overline{\text{STROBE}}$ ceases its normal function and becomes a cycle clock (identical to the F8 multi-chip system write clock, except inverted).

Timing complexities render the capabilities associated with the TEST pin impractical for use in a user application, but these capabilities are sufficient to enable Fairchild to implement a rapid method for thoroughly testing the F3870.

F3870 Clocks
The time bases for the F3870 may originate from one of four external sources; the four external configurations are shown in *Figure 6.* There is an internal 26.5 pF capacitor between XTL_1 and GND, and also between XTL_2 and GND. Thus, external capacitors are not required. In all external clock modes, the external time base frequency is divided by 2 to form the internal ϕ clock.

Fig. 6 F3870 Clock Configurations



Minimum R = 4kΩ	Minimum L = 0.1 mH Minimum Q = 40
C = 20.5 pF ± 2.5 pF + $C_{EXTERNAL}$	Maximum $C_{EXTERNAL}$ = 30 pF
$f_{MIN} \cong \frac{1}{1.1\,RC + 65\text{ ns}}$	C = 10 pF ± 1.3 pF + $C_{EXTERNAL}$
$f_{MAX} \cong \frac{1}{1.0\,RC + 15\text{ ns}}$	$f \cong \frac{1}{2\pi\sqrt{LC}}$
Example with $C_{EXTERNAL} = 0$	Example with $C_{EXTERNAL} = 0$
R = 15 kΩ ± 5% f ≅ 2.9 MHz ± 26%	L = 0.3 mH ± 10% f ≅ 3.0 MHz ± 10%

Instruction Set

The F3870 executes the entire instruction set of the multi-chip F8 family (F3850 family), as shown in *Table 3.* Of course, the STORE instruction is of little use in the F3870 because only read-only memory exists in the addressing range of the data counter (the data counter, however, is incremented if STORE is executed).

A summary of programmable registers and ports is given in *Figure 7.*

Also, for convenient reference, a programming model of the F3870 is given in *Figure 8.*

Table 3 F3870 Instruction Set

Accumulator Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Carry	LNK		ACC←(ACC)+CRY	19	1	1	1/0	1/0	1/0	1/0
Add Immediate	AI	ii	ACC←(ACC)+H'ii'	24 ii	2	2.5	1/0	1/0	1/0	1/0
AND Immediate	NI	ii	ACC←(ACC) Λ H'ii'	21 ii	2	2.5	0	1/0	0	1/0
Clear	CLR		ACC←H'00'	70	1	1				
Compare Immediate	CI	ii	H'ii'+($\overline{ACC}$)+1	25 ii	2	2.5	1/0	1/0	1/0	1/0
Complement	COM		ACC←(ACC)⊕H'FF'	18	1	1	0	1/0	0	1/0
Exclusive OR Immediate	XI	ii	ACC←(ACC)⊕H'ii'	23 ii	2	2.5	0	1/0	0	1/0
Increment	INC		ACC←(ACC)+1	1F	1	1	1/0	1/0	1/0	1/0
Load Immediate	LI	ii	ACC←H'ii'	20 ii	2	2.5	—	—	—	—
Load Immediate Short	LIS	i	ACC←H'0i'	7 i	1	1	—	—	—	—
OR Immediate	OI	ii	ACC←(ACC) V H'ii'	22 ii	2	2.5	0	1/0	0	1/0
Shift Left One	SL	1	SHIFT LEFT 1	13	1	1	0	1/0	0	1/0
Shift Left Four	SL	4	SHIFT LEFT 4	15	1	1	0	1/0	0	1/0
Shift Right One	SR	1	SHIFT RIGHT 1	12	1	1	0	1/0	0	1
Shift Right Four	SR	4	SHIFT RIGHT 4	14	1	1	0	1/0	0	1

Branch Instructions

(In All Conditional Branches, P0 (P0)+2 if the Test Conditions Are Not Met. Execution Is Complete in 30 Cycles.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Branch on Carry	BC	aa	P0←[(P0)+1]+H'aa' if CRY=1	82 aa	2	3/3.5**	—	—	—	—
Branch on Positive	BP	aa	P0←[(P0)+1]+H'aa' if SIGN=1	81 aa	2	3/3.5**	—	—	—	—
Branch on Zero	BZ	aa	P0←[(P0)+1]+H'aa' if ZERO=1	84 aa	2	3/3.5**	—	—	—	—
Branch on True	BT	t,aa	P0←[(P0)+1]+H'aa' if any test is true	8t aa	2	3/3.5**	—	—	—	—
			t=TEST CONDITION							
			2^2 | 2^1 | 2^0							
			ZERO | CRY | SIGN							
Branch if Negative	BM	aa	P0←[(P0)+1]+H'aa' if SIGN=0	91 aa	2	3/3.5**	—	—	—	—
Branch if No Carry	BNC	aa	P0←[(P0)+1]+H'aa' if CARRY≠0	92 aa	2	3/3.5**	—	—	—	—
Branch if No Overflow	BNO	aa	P0←[(P0)+1]+H'aa' if OVF=0	98 aa	2	3/3.5**	—	—	—	—
Branch if Not Zero	BNZ	aa	P0←[(P0)+1]+H'aa' if ZERO=0	94 aa	2	3/3.5**	—	—	—	—
Branch if False Test	BF	t,aa	P0←[(P0)+1]+H'aa' if all false test bits	9t aa	2	3/3.5**	—	—	—	—
			t=TEST CONDITION							
			2^3 | 2^2 | 2^1 | 2^0							
			OVF | ZERO | CRY | SIGN							
Branch if ISAR (Lower) 7	BR7	aa	P0←[(P0)+1]+H'aa' if ISARL≠7	8F aa	2	2.5	—	—	—	—
			P0←(P0)+2 if ISARL=7			2.0	—	—	—	—
Branch Relative	BR	aa	P0←[(P0)+1]+H'aa'	90 aa	2	3.5	—	—	—	—
Jump*	JMP	aaaa	P0←H'aaaa'	29 aaaa	3	5.5	—	—	—	—

Memory Reference Instructions (In All Memory Reference Instructions, the Data Counter Is Incremented DC←DC•1.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Binary	AM		ACC←(ACC)+[(DC)]	88	1	2.5	1/0	1/0	1/0	1/0
Add Decimal	AMD		ACC←(ACC)+[(DC)]	89	1	2.5	1/0	1/0	1/0	1/0
AND	NM		ACC←(ACC) Λ [(DC)]	8A	1	2.5	0	1/0	0	1/0
Compare	CM		[(DC)]+($\overline{ACC}$)+1	8D	1	2.5	1/0	1/0	1/0	1/0
Exclusive OR	XM		ACC←(ACC)⊕[(DC)]	8C	1	2.5	0	1/0	0	1/0
Load	LM		ACC←[(DC)]	16	1	2.5	—	—	—	—
Logical OR	OM		ACC←(ACC) V [(DC)]	8B	1	2.5	0	1/0	0	1/0
Store	ST		(DC)←(ACC)	17	1	2.5	—	—	—	—

*Privileged instruction

** 3.5 cycles if branch taken.

Note

JMP and PI change accumulator contents to the high byte address.

Table 3 F3870 Instruction Set (Cont.)

Address Register Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add to Data Counter	ADC		DC←(DC) + (ACC)	8E	1	2.5	—	—	—	—
Call to Subroutine	PK*		P←(P0); P0U←(r12); PL←(r13)	0C	1	4	—	—	—	—
Call to Subroutine Immediate	PI*	aaaa	P←(P); P0←H'aaaa'‡	28 aaaa	3	6.5	—	—	—	—
Exchange DC	XDC		DC⟷DC1	2C	1	2	—	—	—	—
Load Data Counter	LR	DC,Q	DCU←(r14); DCL←(r15)	0F	1	4	—	—	—	—
Load Data Counter	LR	DC,H	DCU←(r10); DCL←(r11)	10	1	4	—	—	—	—
Load DC Immediate	DCI	aaaa	DC←H'aaaa'	2A aaaa	3	6	—	—	—	—
Load Program Counter	LR	P0,Q	P0U←(r14); P0L←(r15)	0D	1	4	—	—	—	—
Load Stack Register	LR	P,K	PU←(r12); PL←(r13)	09	1	4	—	—	—	—
Return From Subroutine	POP*		P0←(P)	1C	1	2	—	—	—	—
Store Data Counter	LR	Q,DC	r14←(DCU); r15←(DCL)	0E	1	4	—	—	—	—
Store Data Counter	LR	H,DC	r10←(DCU); r11←(DCL)	11	1	4	—	—	—	—
Store Stack Register	LR	K,P	r12←(PU); r13←(P)	08	1	4	—	—	—	—

Scratchpad Register Instructions (Refer to Scratchpad Addressing Modes.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Binary	AS	r	ACC←(ACC) + (r)	Cr	1	1	1/0	1/0	1/0	1/0
Add Decimal	ASD	r	ACC←(ACC) + (r)	Dr	1	2	1/0	1/0	1/0	1/0
Decrement	DS	r	r←(r) + H'FF'	3r	1	1.5	1/0	1/0	1/0	1/0
Load	LR	A,r	ACC←(r)	4r	1	1	—	—	—	—
Load	LR	A,KU	ACC←(r12)	00	1	1	—	—	—	—
Load	LR	A,KL	ACC←(r13)	01	1	1	—	—	—	—
Load	LR	A,QU	ACC←(r14)	02	1	1	—	—	—	—
Load	LR	A,QL	ACC←(r15)	03	1	1	—	—	—	—
Load	LR	r,A	r←(ACC)	5r	1	1	—	—	—	—
Load	LR	KU,A	r12←(ACC)	04	1	1	—	—	—	—
Load	LR	KL,A	r13←(ACC)	05	1	1	—	—	—	—
Load	LR	QU,A	r14←(ACC)	06	1	1	—	—	—	—
Load	LR	QL,A	r15←(ACC)	07	1	1	—	—	—	—
AND	NS	r	ACC←(ACC) ∧ (r)	Fr	1	1	0	1/0	0	1/0
Exclusive OR	XS	r	ACC←(ACC) ⊕ (r)	Er	1	1	0	1/0	0	1/0

Miscellaneous Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Disable Interrupt	DI		RESET ICB	1A	1	2	—	—	—	—
Enable Interrupt*	EI		SET ICB	1B	1	2	—	—	—	—
Input	IN	aa	ACC←(INPUT PORT aa)	26 aa	2	4	0	1/0	0	1/0
Input Short	INS	a	ACC←(INPUT PORT a)	Aa	1	4***	0	1/0	0	1/0
Load ISAR	LR	IS,A	ISAR←(ACC)	0B	1	1	—	—	—	—
Load ISAR Lower	LISL	a	ISARL←a	01101a**	1	1	—	—	—	—
Load ISAR Upper	LISU	a	ISARU←a	01100a**	1	1	—	—	—	—
Load Status Register*	LR	W,J	W←(r9)	1D	1	2	1/0	1/0	1/0	1/0
No-Operation	NOP		P0←(P0) + 1	2B	1	1	—	—	—	—
Output	OUT	aa	OUTPUT PORT aa←(ACC)	27 aa	2	4	—	—	—	—
Output Short	OUTS	a	OUTPUT PORT a←(ACC)	Ba	1	4***	—	—	—	—
Store ISAR	LR	A,IS	ACC←(ISAR)	0A	1	1	—	—	—	—
Store Status Register	LR	J,W	r9←(W)	1E	1	1	—	—	—	—

*Privileged instruction
**3-bit octal digit
***Two machine cycles for CPU ports
‡Contents of ACC destroyed

Table 3 F3870 Instruction Set (Cont.)

Notes
Each lower case character represents a hexadecimal digit.
Each cycle equals four machine clock periods.
Lower case denotes variables specified by the programmer.

Function Definitons

←	is replaced by
( )	the contents of
(—)	binary ones complement of
+	arithmetic add (binary or decimal)
⊕	logical OR exclusive
Λ	logical AND
V	logical OR inclusive
H'#'	hexadecimal digit

Register Names

a	address variable
A	accumulator
DC	data counter (indirect address register)
DC1	data counter #1 (auxiliary data counter)
DCL	least significant eight bits of data counter addressed
DCU	most significant eight bits of data counter addressed
H	scratchpad register #10 and #11
i and ii	immediate operand
ICB	interrupt control bit
IS	indirect scratchpad address register
ISAR	indirect scratchpad address register
ISARL	least significant three bits of ISAR
ISARU	most significant three bits of ISAR
J	scratchpad register #9
K	registers #12 and #13
KL	register #13
KU	register #12
P0	program counter
P0L	least significant eight bits of program counter
P0U	most significant eight bits of program counter
P	stack register
PL	least significant eight bits of program counter
PU	most significant eight bits of active stack register
Q	registers #14 and #15
QL	register #15
QU	register #14
r	scratchpad register (any address through 11)
W	status register

Scratchpad Addressing Modes (Machine Code Format)

r = C	(hexadecimal) register addressed by ISAR (unmodified)
r = D	(hexadecimal) register addressed by ISAR; ISARL incremented
r = E	(hexadecimal) register addressed by ISAR; ISARL decremented
r = F	(no operation performed)
r = 0-B	(hexadecimal) register 0 through 11 addressed directly from the instruction

Status Register

—	no change in condition
1/0	is set to 1 or 0, depending on conditions
CRY	carry flag

Mask Options

The ROM array may contain object program code and/or tables of nonvarying data. Every F3870 is implemented using a custom mask that specifies the state of every ROM bit, as well as certain address mask options that are external to the ROM array. The following mask options are specified:

1. The 1024, 2048, 3072, or 4096 bytes of ROM storage. This reflects programs and permanent data tables stored in the PSU memory.
2. Input/output ports can be any of the following three configurations:
   a. Standard pull-up
   b. Open drain
   c. Direct drive
3. Input/output ports 0 and 1 can be specified either cleared or unaltered following an external reset.
4. External interrupt and external reset can be specified to have or omit an internal pull-up resistor.
5. The I/O port output option choices are: the standard pull-up (option A), the open drain (option B), and the driver pull-up (option C).

The format for mask options must be submitted to Fairchild Microprocessor Division before device manufacture. The data to be stored in permanent memory may be submitted in the form of an EPROM or HP2644/HP2645 cartridge (Formulator format only). Other options must be specified on the Fairchild ROM Code Entry Form, available from a Fairchild representative.

Fig. 7 Programmable Registers and Ports

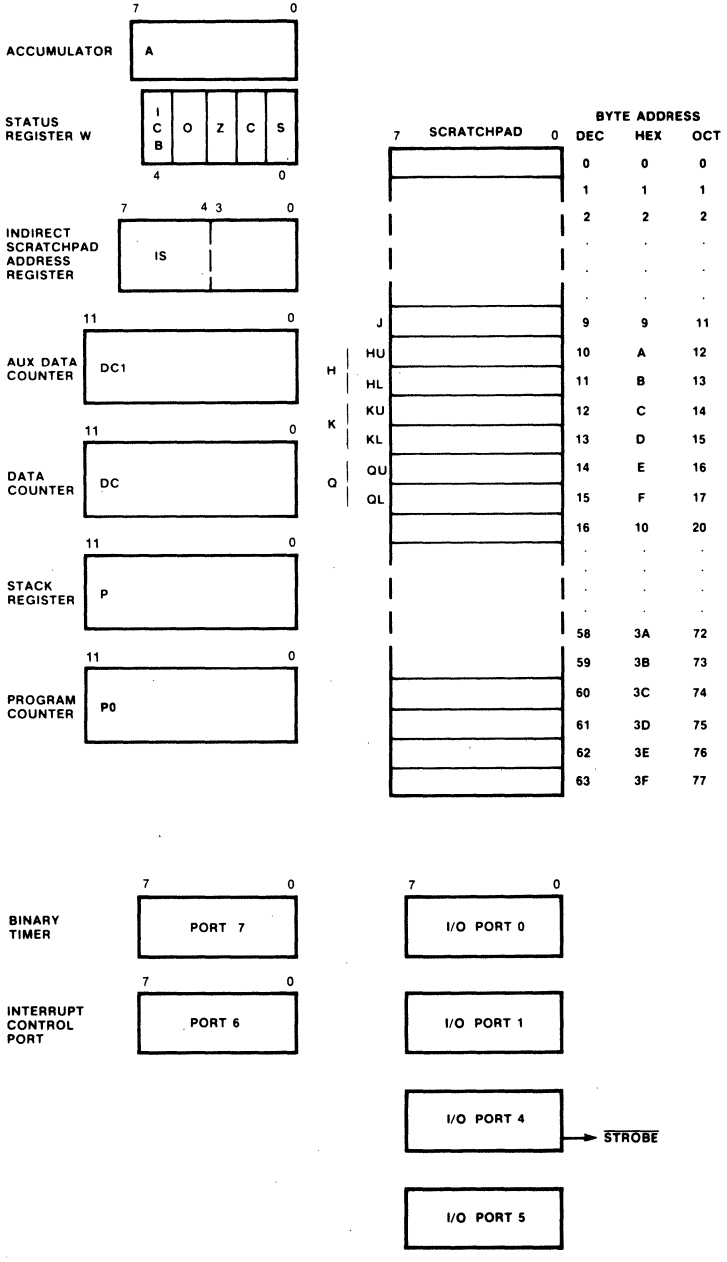
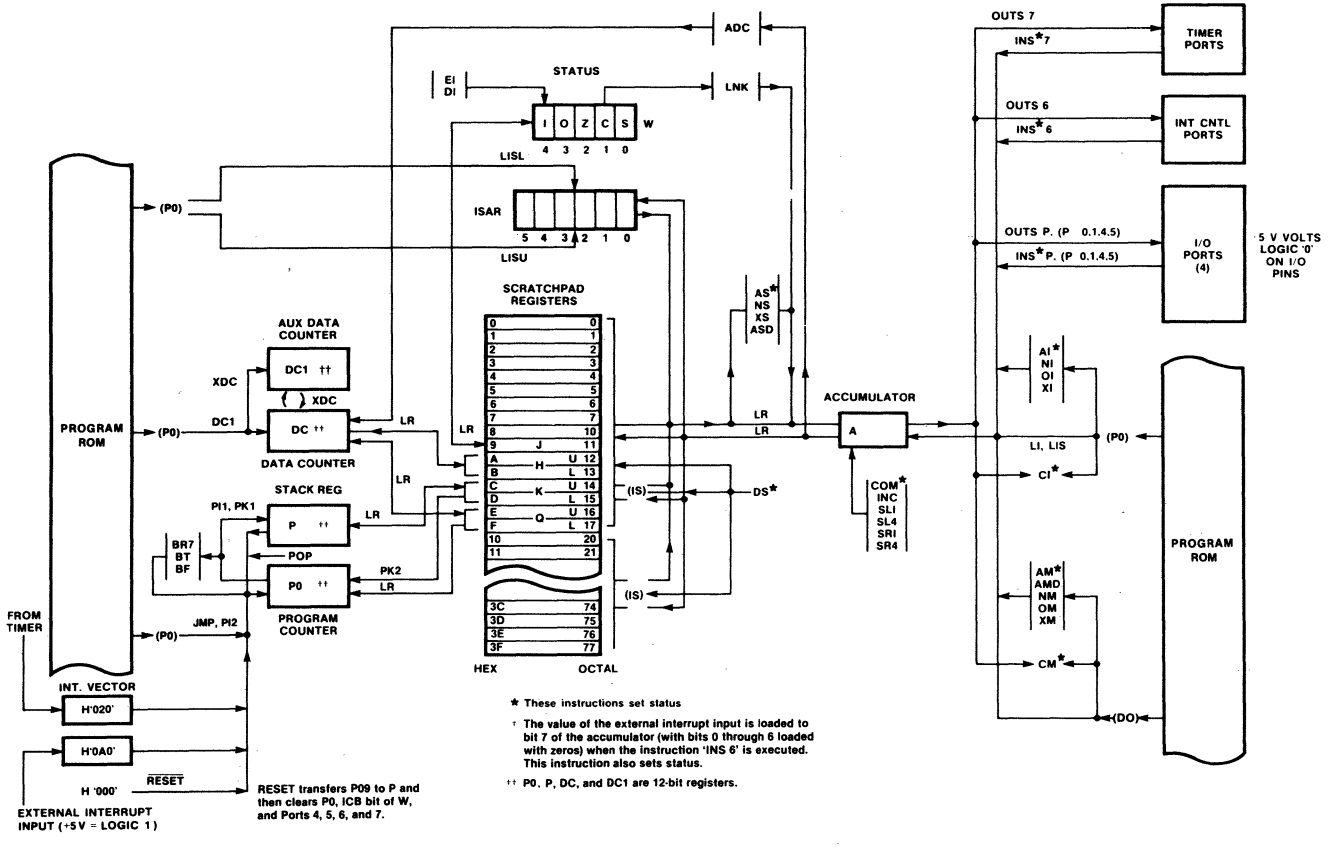


Fig. 8 Programming Model



* These instructions set status

† The value of the external interrupt input is loaded to bit 7 of the accumulator (with bits 0 through 6 loaded with zeros) when the instruction 'INS 6' is executed. This instruction also sets status.

†† P0, P, DC, and DC1 are 12-bit registers.

Note:
The instructions PI and PK are shown in two sequential parts (PI1, PI2, and PK1, PK2).

Supplementary Notes

For total software compatibility when expanding into a multi-chip configuration, the F3871 Peripheral Input/Output circuit should be used. The F3871 has the same improved timer (binary count, readable, and three modes of operation) and ready strobe outputs as the F3870.

The interrupt control bit of the status register is automatically reset when an interrupt request is acknowledged. It is then the programmer's responsibility to determine when the ICB is again to be set (by executing the E1 instruction). This action prevents an interrupt service routine from being interrupted unless the programmer so desires.

When reading the interrupt control port (port 6), bit 7 of the accumulator is loaded with the actual logic level being applied to the EXT INT pin, regardless of the status of ICP bit 2 (the EXT INT active level bit); that is, if the EXT INT pin is at +5 V, bit 7 of the accumulator is set to a logic 1, but if the EXT INT pin is at ground, accumulator bit 7 is reset to logic 0.

In *Table 3,* the number of cycles shown is "nominal machine cycles." A nominal machine cycle is defined as 4 ϕ clock periods, thus requiring 2 μs for a 2 MHz clock frequency (4 MHz external time base frequency).

Table 3 also uses the following nomenclature for register names:

F8 ⟷ F3870
PC_0 = P0 Program Counter
PC_1 = P Stack Register
DC_0 = DC Data Counter
DC_1 = DC1 Auxiliary Data Counter

This nomenclature is used to be consistent with the assembly language mnemonics.

For the F3870, execution of an INS or OUTS instruction requires two machine cycles for ports 0 and 1, whereas ports 4 and 5 require four machine cycles.

When an external reset of the F3870 occurs, P0 pushes into P and the old contents of P are lost. It must be noted that an external reset is recognized at the start of the machine cycle and not necessarily at the end of an instruction. Thus, if the F3870 is executing a multi-cycle instruction, that instruction is not completed and the contents of P upon reset may not necessarily be the address of the instruction that would have been executed next. It may, for example, point to an immediate operand if the reset occurred during the second cycle of an L1 or C1 instruction. Additionally, several instructions (JMP, P1, PK, LR, P0 and Q) as well as the interrupt acknowledge sequence modify P0 in parts. That is, they alter P0 by loading first one part, then the other, and the entire operation takes more than one cycle. Should reset occur during this modification process, the value pushed into P is part of the old P0 (the as-yet unmodified part) and part of the new P0 (already-modified part). Thus, care should be taken (perhaps by external gating) to ensure that reset does not occur at an undesirable time if any significance is to be given to the contents of P after a reset occurs.

Timing Characteristics

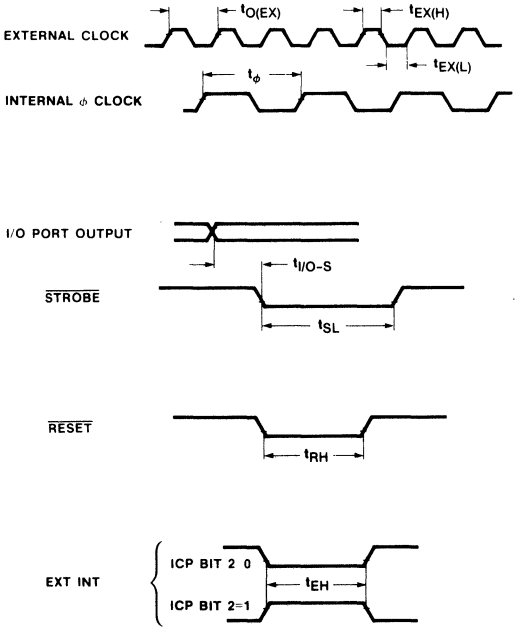
The F3870 timing characteristics are described in *Table 4* and illustrated in *Figures 9* and *10.*

Table 4 Timing Characteristics

Signal	Symbol	Characteristic	Min	Max	Unit	Notes
XTL_1 XTL_2	$t_o(EX)$	Time Base Period, All External Modes	250	1000	ns	4 MHz-1 MHz
	$t_{EX}(H)$	External Clock Pulse Width, high	90	700	ns	
	$t_{EX}(L)$	External Clock Pulse Width, low	100	700	ns	
ϕ	t_ϕ	Internal ϕ Clock Period	$2t_\phi$			
WRITE	t_W	Internal WRITE Clock Period	$4t_\phi$ $6t_\phi$			Short Cycle Long Cycle
I/O	$t_{dI/O}$	Output Delay from Internal WRITE Clock	0	1000	ns	50 pF Plus One TTL Load
	$t_{sI/O}$	Input Setup Time to WRITE Clock	1000		ns	
$\overline{STROBE}$	$t_{I/OS}$	Output Valid to $\overline{STROBE}$ Delay	$3t_\phi$ -1000	$3t_\phi$ $+250$	ns	Note 1
	t_{SL}	$\overline{STROBE}$ low Time	$8t_\phi$ -250	$12t_\phi$ $+250$	ns	
$\overline{RESET}$	t_{RH}	$\overline{RESET}$ Hold Time, low	$6t_\phi$ $+750$		ns	
EXT INT	t_{EH}	EXT INT Hold Time, Active State	$6t_\phi$ $+750$		ns	To Trigger Interrupt
		EXT INT Hold Time, Inactive State	$2t_\phi$		ns	To Trigger Timer; Note 2
	C_{IN}	Input Capacitance: I/O Ports, $\overline{RESET}$, EXT INT		7	pF	Unmeasured Pins Returned to V_{SS}; Note 4
	C_{XTL}	Input Capacitance: XTL_1, XTL_2	23.5	29.5	pF	Unmeasured Pins Returned to V_{SS}; Note 4

Notes
1. I/O load is 50 pF plus one standard TTL input; $\overline{STROBE}$ load is 50 pF plus three standard TTL inputs.
2. Specification is applicable when the timer is in the interval timer mode.
3. $T_A = 0°C$ to $+70°C$, $V_{DD} = +5\ V \pm 10\%$, I/O power dissipation $\leq$ 100 mW, unless otherwise noted.
4. $T_A = 25°C$, f = 2 MHz.

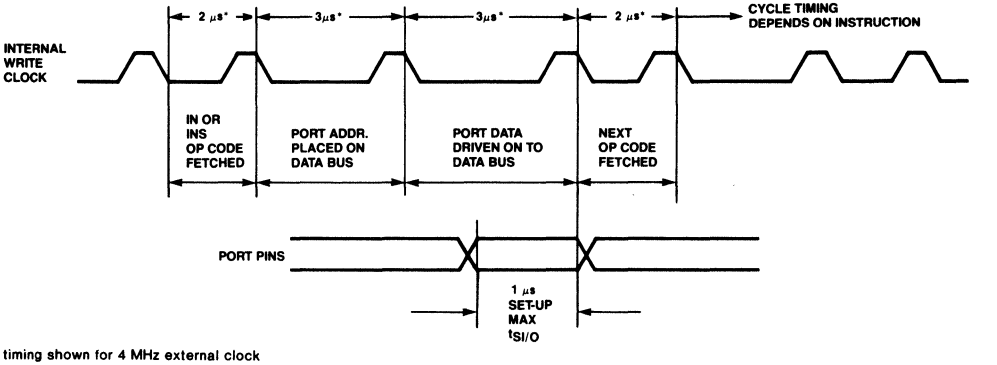
Fig. 9 Timing Diagrams



Note
All measurements are referenced to V_{IL} max, V_{IH} min, V_{OL} max, or V_{OH} min

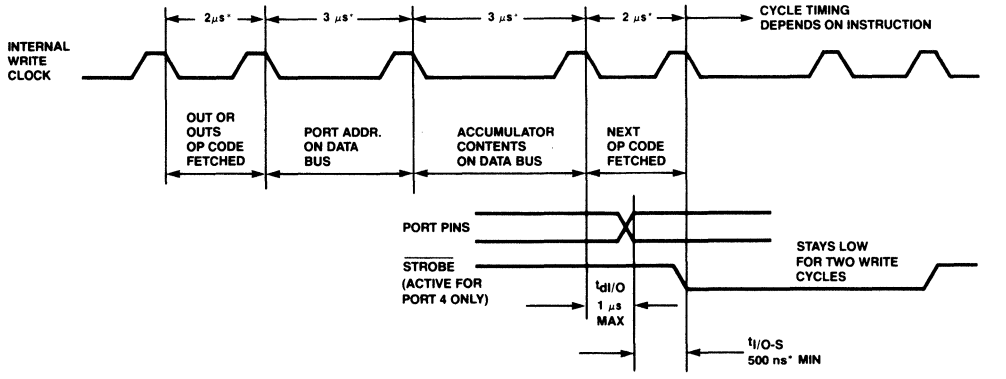
Fig. 10 Port Input/Output Timing Diagrams

A. Input on Port 4 or 5



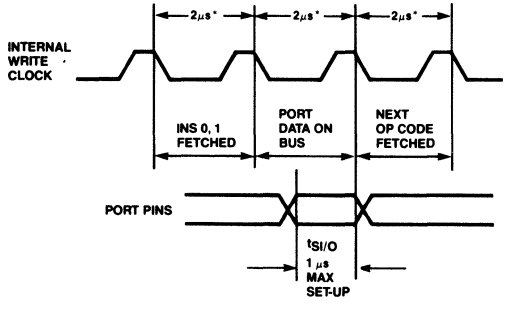
*Cycle timing shown for 4 MHz external clock

B. Output on Port 4 or 5



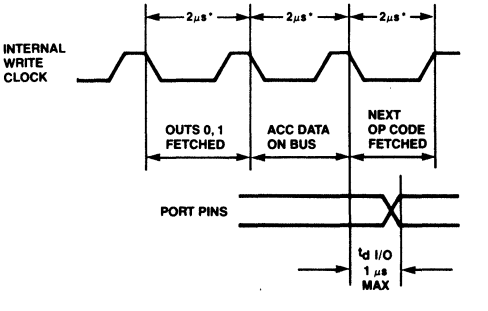
*Cycle timing shown for 4 MHz external clock

C. Input on Port 0 or 1



*Cycle timing shown for 4 MHz external clock

D. Output on Port 0 or 1



*Cycle timing shown for 4 MHz external clock

DC Characteristics

The dc characteristics of the F3870 are described in *Table 5.*

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Voltage on any Pin with Respect to Ground (Except Open-Drain Pins)	− 1.0 V, + 7 V
Voltage on any Open-Drain Pin	− 1.0 V, + 13.2 V
Power Dissipation	1.5 W
Ambient Temperature Under Bias	0°C, + 70 °C
Storage Temperature	− 55°C, + 150°C

Table 5 DC Characteristics $T_A = 0°C$ to $+70°C$, $V_{CC} = +5\ V \pm 10\%$, I/O power dissipation ≤ 100 mW

Symbol	Characteristic	Min	Max	Unit	Conditions
I_{CC}	Power Supply Current		100	mA	Outputs Open
P_D	Power Dissipation		550	mW	Outputs Open
V_{IHEX}	External Clock Input HIGH Voltage	2.4	5.8	V	
V_{ILHEX}	External Clock Input LOW Voltage	− 0.3	0.6	V	
I_{HEX}	External Clock Input HIGH Current		100	μA	$V_{IHEX} = 2.4$ V
I_{ILEX}	External Clock Input LOW Current		− 100	μA	$V_{ILEX} = 0.6$ V
V_{IH}	Input HIGH Voltage	2.0	5.8	V	
V_{IL}	Input LOW Voltage	− 0.3	0.8	V	
I_{IH}	Input HIGH Current (Except Open-Drain and Direct-Drive I/O Ports)		100	μA	$V_{IH} = 2.4$ V, Internal Pull-Up
I_{IL}	Input LOW Current (Except Open-Drain and Direct-Drive Ports)		− 1.6	mA	$V_{IL} = 0.4$ V
I_{LOD}	Leakage Current (Open-Drain Ports)		± 10	μA	Pull-Down, Device Off, $V_{OH} = 13.2$ V
I_{OH}	Output HIGH Current (Except Open-Drain and Direct-Drive Ports)	− 100		μA	$V_{OH} = 2.4$ V
I_{OHDD}	Output Drive Current (Direct-Drive Ports)	− 1.5	− 8.0	mA	$V_{OH} = 0.7$ V to 1.5 V
I_{OL}	Output LOW Current	1.8		mA	$V_{OL} = 0.4$ V
I_{OHS}	Output HIGH Current (STROBE Output)	− 300		μA	$V_{OH} = 2.4$ V
I_{OLS}	Output LOW Current (STROBE Output)	5.0		mA	$V_{OL} = 0.4$ V

Ordering Information

Order Code	Package	Temperature Range*
F3870DC	Ceramic	C
F3870DL	Ceramic	L
F3870DM	Ceramic	M
F3870PC	Plastic	C

*C = Commercial Temperature Range 0°C to +70°C
L = Limited Temperature Range −40°C to +85°C
M = Military Temperature Range −55°C to +125°C

FAIRCHILD
A Schlumberger Company

F3870A/F3870B High-Speed Single-Chip Microcomputer

Advance Product Information

Microprocessor Product

Description

The Fairchild F3870A and F3870B are advancements in the F3870 series of single-chip microcomputers. The F3870A and F3870B offer higher instruction execution speed, thereby improving the throughput of the microcomputer.

- **Fully Hardware- and Software- Compatible with the F3870 Series of Microcomputers**
- **The F3870A Offers An Instruction Cycle Time of 1.33 μsec, and the F3870B Cycle Time is 1 μsec.**
- **Mask Option Internal Clock Divider**

	Clock Crystal Frequency		
	Without	With	Cycle
Unit	Internal ÷ 2	Internal ÷ 2	Time
F3870		4 MHz	2 μs
F3870A	3 MHz	6 MHz	1.33 μs
F3870B	4 MHz	8 MHz	1 μs

For additional information, see the F3870 data sheet

FAIRCHILD
A Schlumberger Company

F38C70 Single-Chip Microcomputer

Microprocessor Product

Description

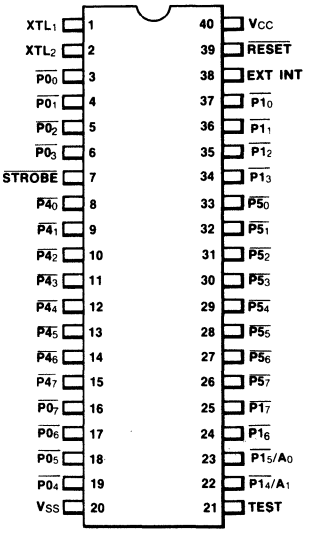
The Fairchild F38C70 8-bit single-chip microcomputer is a member of the F387X series; it executes all of the F8 instruction set and is software-compatible with the F3870. Additional power-save instructions provide two different power-save modes.

Implemented in ion-implanted CMOS doublepoly silicon-gate technology, the F38C70 offers maximum cost effectiveness in a wide range of applications requiring very low power consumption.

More than 70 commands of the F8 instruction set are executed by the single-chip microcomputer, which features 2048 bytes of ROM, 64 bytes of scratchpad RAM, a programmable timer, 32 bits of I/O, and a single +5 V power supply.

- **Single CMOS Integrated Circuit**
- **Software-Compatible with F8 and F3870**
- **2048-Byte Mask Programmable ROM**
- **64-Byte Scratchpad RAM**
- **32-Bit I/O with Four Options**
- **8-Bit Programmable Timer with 16-Bit Programmable Prescaler**
- **External Interrupt**
- **Crystal, LC, RC, or External Clock**
- **Single +5 V (±10%) Power Supply**
- **Power-Save (PS) and Power-Save All (PSA) Modes**
- **Option for all Short Machine Cycles**
- **Direct Replacement for F3870**
- **Low Power (50 mW typ., 5 mW in PS mode, 0.5 mW in PSA mode)**

Connection Diagram
40-Pin DIP



(Top View)

Signal Functions

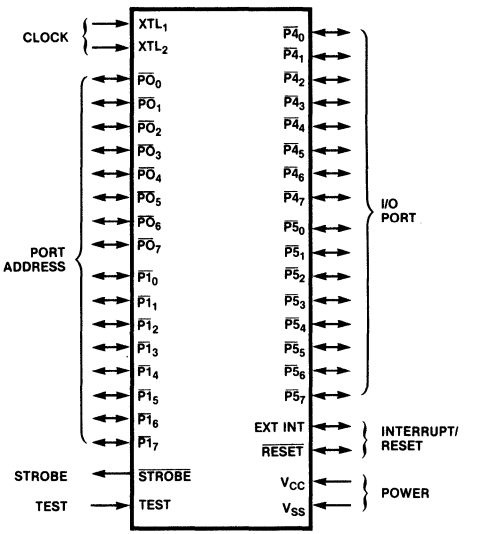
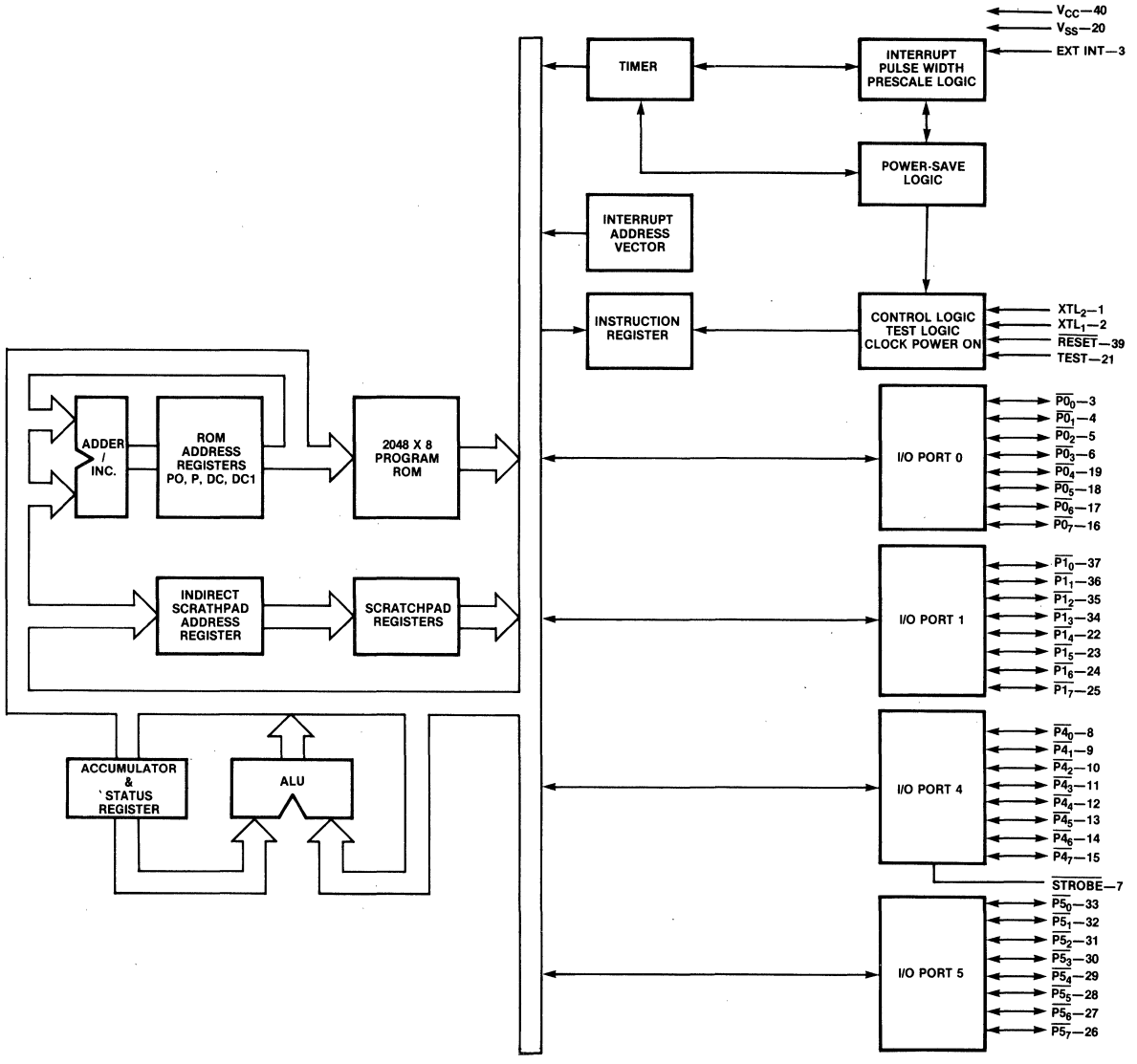


Figure 1 Block Diagram



Main Control Logic

The instruction register (IR) receives the operation code (OP code) of the instruction to be executed from the program ROM through the data bus. Eight bits are latched into the IR during all OP code fetches. Some instructions are completely specified by the upper four bits of the OP code; in these instructions, the lower four bits are an immediate register address or an immediate 4-bit operand. Once latched into the IR, the main control logic decodes the instruction and provides the necessary control gating signals to all circuit elements.

ROM Address Registers

Four 12-bit registers are associated with the program ROM: program counter PO, stack register P, data counter DC0, and auxiliary data counter DC1. The program counter is used to address instructions or immediate operands; the stack register is used to save the contents of PO during an interrupt or subroutine call. Thus, P contains the return address at which processing is to resume upon completion of the subroutine or the interrupt routine.

The data counter is used to address data tables. This register is autoincrementing. Of the two data counters, only DC0 can access the ROM; however, the XDC instruction allows DC0 and DC1 to be exchanged.

Associated with the address registers is a 12-bit adder/incrementer. This logic element is used to increment PO or DC when required and to add displacements to PO on relative branches or to add the data bus contents to DC0 in the add data counter (ADC) instruction.

Program ROM

The microcomputer program and data constants are stored in the 2048 X 8 byte program ROM. When a ROM access is required, the appropriate address register (PO or DC0) is gated onto the ROM address bus and the ROM output is gated onto the main data bus. The first byte in the ROM is location zero.

Scratchpad and ISAR

The scratchpad provides 64 8-bit registers that can be used as general purpose RAM memory. The indirect scratchpad address register (ISAR) is a 6-bit register used to address the 64 registers. All 64 registers can be accessed using the ISAR. In addition, the lower order 12 registers can also be directly addressed.

The ISAR can be visualized as holding two octal digits. This division of the ISAR is important, since a number of instructions increment or decrement only the least significant three bits of the ISAR when referencing scratchpad bytes through the ISAR. This simplifies referencing a buffer of contiguous scratchpad bytes. For example, when the low-order octal digit is incremented or decremented, the ISAR is incremented from octal 27 (0'27) to 0'20' or is decremented from 0'20' to 0'27'. This feature of the ISAR is very useful in many program sequences.

All six bits of the ISAR can be loaded at one time, or either half can be loaded independently.

The decimal scratchpad registers (9 through 15) are given mnemonic names (J, H, K, and Q) because of special linkages between these and other registers, such as the stack register. These special linkages simplify the performance of multi-level interrupts and subroutine nesting. For example, the instruction LR K,P stores the lower eight bits of the stack register into register 13 (K lower, or KL) and stores the upper three bits of P into register 12 (K upper, or KU).

Arithmetic and Logic Unit (ALU)

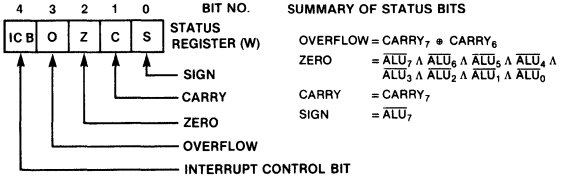
After receiving commands from the main control logic, the ALU performs the required arithmetic or logic operations (using the data presented on the two input buses) and provides the result on the result bus. The arithmetic operations performed in the ALU are binary add, decimal adjust, add with carry, decrement, and increment. The logic operations performed are AND, OR, exclusive-OR, ones complement, shift right, and shift left. The ALU also provides four signals presenting the status of the result. These signals, stored in status register W, represent the carry, overflow, sign, and zero condition of the operation.

Accumulator

The accumulator (ACC) is the prinicpal register for data manipulation within the F38C70. The ACC serves as one input to the ALU for arithmetic or logic operations; the results of ALU operations are stored in the ACC.

Status Register

The status (W) register holds five status flags:



Interrupt Control Bit

The interrupt control bit (ICB) is used to allow or disallow interrupts in the F38C70. (This bit is not the same as the two interrupt enable bits in the interrupt control port.) If the ICB is set and the F38C70 interrupt logic communicates an interrupt request to the CPU section, the interrupt is acknowledged and processed upon completion of the first non-privileged instruction. If the ICB is cleared, an interrupt request is not acknowledged or processed until the ICB is set again.

I/O Ports

The F38C70 provides four complete bidirectional input/output ports: 0, 1, 4, and 5. In addition, the interrupt control port is addressed as port 6, and the binary timer is addressed as port 7. Ports 8 and 9 are the 16-bit holding register for the timer prescaler.

An output instruction (OUT or OUTS) causes the contents of the ACC to be latched into the addressed port. An input instruction (IN or INS) transfers the contents of the port to the ACC (port 6, an exception, is described in the "Timer and Interrupt Control Port") section. The I/O buffers on the F38C70 are logically inverted.

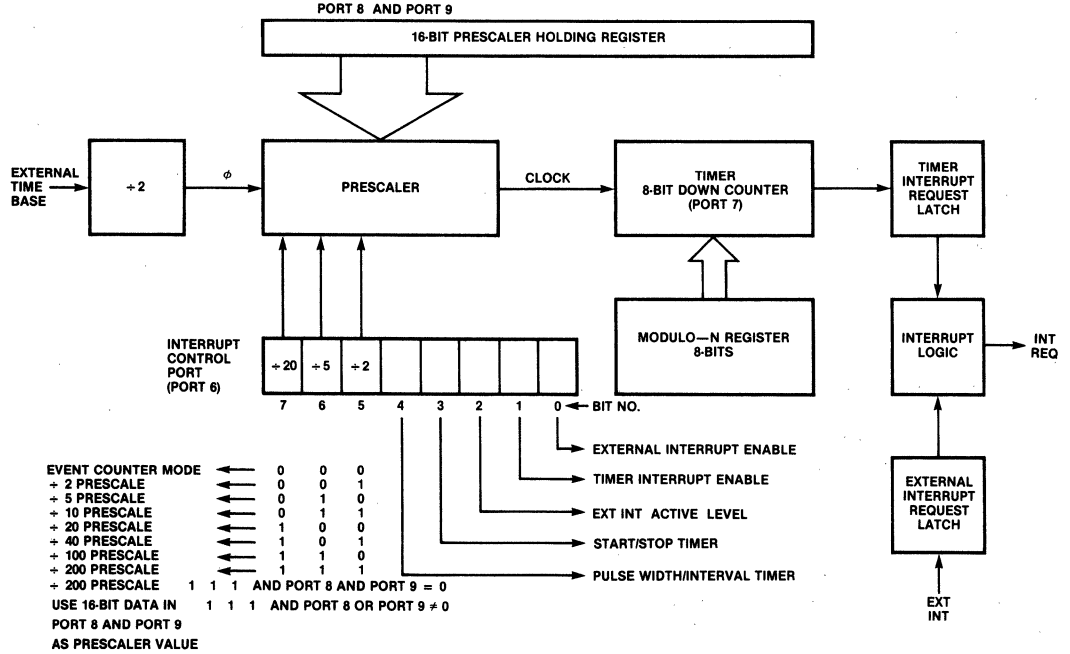
An output ready strobe is associated with port 4. This flag is used to signal a peripheral device that the F38C70 has just completed an output of new data to port 4. Because the strobe provides a single low pulse shortly after the output operation is complete, either edge can be used to signal the peripheral. The STROBE signal is also used to request new input information from a peripheral by performing a dummy output of H'00' to port 4 after completing the input operation.

Four output drive options are available for the F38C70 I/O ports. Individual bits of the four I/O ports are configured as

1. Open drain
2. CMOS 3-state push-pull buffer
3. TTL-compatible
4. CMOS push-pull buffer

For the 3-state push-pull buffer, the I/O pin goes 3-state when executing an INS instruction to that port and remains in 3-state until an OUTS instruction is executed to that port

Figure 2. Timer and Interrupt Control Port Block Diagram



Timer and Interrupt Control Port
The timer is an 8-bit binary down counter that is software-programmable to operate in one of three modes: interval timer, pulse width measurement, or event counter. As shown in figure 2, an 8-bit register (interrupt control port), a programmable 16-bit prescaler, and an 8-bit modulo-N register are associated with the timer.

The timer mode, prescale value, timer start and stop, active level of the EXT INT pin, and interrupt local enable/disable are selected by the proper bit configuration output from the accumulator to interrupt control port 6 with an OUT or OUTS instruction. Bits within the interrupt control port are defined as follows:

Bit 0 = External interrupt enable
Bit 1 = Timer interrupt enable
Bit 2 = EXT INT active level
Bit 3 = Start/stop timer
Bit 4 = Pulse width/internal timer
Bit 5 = ÷2 Prescaler control
Bit 6 = ÷5 Prescaler control
Bit 7 = ÷20 Prescaler control

Timer

The F38C70 timer, like the F3870, is an 8-bit programmable down counter. However, the F38C70 has two additional 8-bit registers (ports 8 and 9) that can be accessed by output instructions. These registers can be used to generate very long interval timer interrupts or any desired prescaler value.

A special situation exists when reading the interrupt control port with an IN or INS instruction). The accumulator is *not* loaded with the content of the ICP; instead, accumlator bits 0 through 6 are loaded with zeros, and bit 7 is loaded with the logic level being applied to the EXT INT pin. Thus, the status of EXT INT can be determined without needing to service an external interrupt request. This capability is useful in establishing a high-speed polled handshake procedure or for using EXT INT as an extra input pin if external interrupts are not required and the timer is used only in the interval timer mode.

The rate at which the timer is clocked in the interval timer mode is determined by the frequency of an internal Φ clock and by the division value selected for the prescaler. (The internal Φ clock operates at one-half the external time base frequency.) Assuming ports 8 and 9 have been loaded with zeros, if ICP bit 5 is set and bits 6 and 7 are cleared, the prescaler divides Φ by two. In the same manner, if bit 6 or 7 is individually set, the prescaler divides Φ by 5 or 20, respectively. Combinations of bits 5, 6, and 7 may also be selected. For example, if bits 5 and 7 are set, while 6 is cleared, the prescaler will divide by 40. Thus, possible prescaler values are ÷2, ÷5, ÷10, ÷20, ÷40, ÷100, and ÷200. If bits 5, 6, and 7 of the interrupt control port are set, and the contents of either of the two prescaler registers are not zero, the timer uses the value that is held in the two registers as a 16-bit prescaler value.

Any of three conditions will cause the prescaler to be reset:

1. When the timer is stopped by clearing ICP bit 3
2. When an output instruction to port 7 (the timer is assigned Port Address 7) is executed
3. On the trailing edge transition of the EXT INT pin when in the pulse width measurement mode

An OUT or OUTS instruction to port 7 loads the contents of the accumulator to both the timer and the 8-bit modulo-N register, resets the prescaler, and clears any previously stored timer interrupt request. The timer is an 8-bit down-counter clocked by the prescaler in both the interval timer mode and the pulse width measurement mode. The prescaler is not used in the event counter mode. The modulo-N register is used as a buffer in all three timer modes. Its function is to save the value that was most recently output to port 7.

Interval Timer Mode
When ICP bit 4 is cleared (logic 0) and at least one prescale bit is set, the timer operates in the interval timer mode. When bit 3 of the ICP is set, the timer starts counting down from the modulo-N value. After counting down to H'01', the timer returns to the modulo-N value at the next count. On the transition from H'01' to H'N', the timer sets a timer interrupt request latch. Note that the interrupt request latch is set by the transition of H'N' in the timer, thus allowing a full 256 counts if the modulo-N register is preset to H'00'.

If bit 1 of the ICP is set, the interrupt request is passed on to the CPU section of the F38C70. However, if bit 1 of the ICP is a logic 0, the interrupt request is not passed on the the CPU section, although the interrupt request latch remains set. If ICP bit 1 is subsequently set, the interrupt request is then passed on to the CPU section. (The interrupt request is acknowledged by the CPU section only if ICB is set.) Only two events reset the timer interrupt request latch: the timer interrupt request is acknowledged by the CPU section, or a new load of the modulo-N register is performed.

If the modulo-N register is loaded with H'64' (decimal 100), the timer interrupt request latch is set at the 100th count following the timer start and the latch is repeatedly set on precise 100-count intervals. If the prescaler is set at ÷40, the timer interrupt request latch is set every 4000 Φ clock

periods. For a 2-mHz Φ clock (4-mHz time base frequency), this produces 2 ms intervals.

If ports 8 and 9 are loaded with zeros, the range of possible intervals is from 2 to 51,200 Φ clock periods (1 μs to 25.6 ms for a 2-mHz Φ clock). However, approximately 50 Φ periods is a practical minimum, because the time between setting the interrupt request latch and the execution of the first instruction of the interrupt service routine is at least 29 Φ periods (the response time is dependent on how many privileged instructions are encountered when the request occurs).

To establish time intervals greater than 51,200 Φ clock periods, the 16-bit prescaler or the timer interrupt service routine can be used to count the number of interrupts, saving the result in one or more of the scratchpad registers until the desired interval is achieved. Virtually any time interval, or several time intervals, can be generated using this technique.

The timer is read at any time and in any mode, using an input instruction (IN 7 or INS 7), and can take place "on-the-fly" without interfering in normal timer operation. Also, the timer can be stopped at any time by clearing bit 3 of the ICP. The timer holds its current contents indefinitely and resumes counting when bit 3 is set again. The prescaler is reset whenever the timer is stopped; thus, a series of starting and stopping results in a cumulative truncation error.

For a free-running timer in the interval timer mode, the time interval between any two interrupt requests can be in error by ±6 Φ clock periods, although the cumulative error over many intervals is zero. The prescaler and timer generate precise intervals for setting the timer interrupt request latch, but the time out can occur at any time within a machine cycle. (There are two machine cycle types: short, which consist of 4 Φ clock periods, and long, which consist of 6 Φ clock periods.) The Fairchild multi-chip F8 family has a write clock signal that corresponds to a machine cycle. Interrupt requests are synchronized with the internal write clock, thus providing the possible ±6 Φ error. Additional errors may arise if the interrupt request occurs while a privileged instruction or multi-cycle instruction is being executed. Nevertheless, for most applications, all the above errors are negligible, especially if the desired time interval is greater than one ms.

Pulse Width Measurement Mode

When ICP bit 4 is set (logic 1) and at least one prescale bit is set, the timer operates in the pulse width measurement mode. This mode is used to accurately measure the duration of a pulse applied to the EXT INT pin. The timer is stopped and the prescaler is reset whenever EXT INT is at its inactive level. The active level of EXT INT is defined by ICP bit 2: if cleared, EXT INT is active low; if set, EXT INT is active high.

If ICP bit 3 is set, the prescaler and timer start counting when EXT INT transfers to the active level. When EXT INT returns to the inactive level, the timer stops, the prescaler resets, and, if ICP bit 0 is set, an external interrupt request latch is set. (Unlike timer interrupts, external interrupts are not latched if the ICP interrupt enable bit is not set.)

As in the interval timer mode, the timer can be read at any time and can be stopped at any time by clearing ICP bit 3 (the prescaler and ICP bit 1 function as described in the interval timer mode section). The timer still functions as an 8-bit binary down counter with the interrupt request latch set on the timer's transition from H'01' to H'N' (modulo-N value). Note that the EXT INT pin has nothing to do with loading the timer; its action is that of automatically starting and stopping the timer and of generating external interrupts. Pulse widths longer than the prescale value times the modulo-N value are easily measured by using the timer interrupt service routine to store the number of timer interrupts in one or more scratchpad registers.

The actual pulse duration is typically slightly longer than the measured value, because the prescaler status is not readable and is reset when the timer is stopped. Thus, for maximum accuracy, it is advisable to use a small division setting for the prescaler.

Event Counter Mode

When ICP bit 4 is cleared and all prescale bits (ICP bits 5, 6, and 7) are cleared, the timer operates in the event counter mode. This mode is used for counting pulses applied to the EXT INT pin. If ICP bit 3 is set, the timer will decrement on each transition from the inactive level to the active level of the EXT INT pin. The prescaler is not used in this mode. As in the other two timer modes, the timer can be read at any time and can be stopped at any time by clearing ICP bit 3, ICP bit 1 functions as previously described, and the timer interrupt request latch is set on the timer's transition from H'01' to H'N' (modulo-N value).

Normally, ICP bit 0 should be kept cleared in the event counter mode; otherwise, external interrupts are generated on the transition from the inactive level to the active level of the EXT INT pin.

For the event counter mode, the minimum pulse width required on EXT INT is 2 Φ clock periods and the minimum inactive time is 2 Φ clock periods; therefore, the maximum repetition rate is 500 Hz.

External Interrupts
When the timer is in the interval timer mode, the EXT INT pin is available for non-timer related interrupts. If ICP bit 0 is set, an external interrupt request latch is set for a transition from the inactive level to the active level of EXT INT. (The EXT INT signal is an edge-triggered input.) The interrupt request is latched either until acknowledged by the CPU section or until ICP bit 0 is cleared (unlike timer interrupt requests that remain latched even when ICP bit 1 is cleared).

External interrupts are handled in the same fashion when the timer is in the pulse width measurement mode or in the event counter mode, except that when in the pulse width measurement mode, the external interrupt request latch is set on the trailing edge of EXT INT (that is, on the transition from the active level to the inactive level).

Interrupt Handling
When either a timer or an external interrupt request is communicated to the CPU section of the F38C70, it is acknowledged and processed at the completion of the first non-privileged instruction if the interrupt control bit of the status register is set. If the interrupt control bit is not set, the interrupt request continues either until the interrupt control bit is set and the CPU section acknowledges the interrupt or until the interrupt request is cleared (as previously described).

If a timer interrupt request and an external interrupt request occur simultaneously, when the CPU section starts to process the requests, the timer interrupt is handled first.

When an interrupt is allowed, the CPU section requests that the interrupting element pass its interrupt vector address to the program counter through the data bus. The vector address for a timer interrupt is H'020'. The vector address for external interrupts is H'0A0'. After the vector address is passed to the program counter, the CPU section sends an acknowledge signal to the appropriate interrupt request latch, which clears that latch. The interrupt service routine executes; the return address of the original program is automatically stored in stack register P.

Power-On Clear

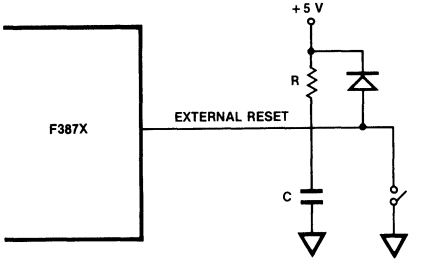
The F38C70 contains power-on clear circuitry to automatically reset the internal logic following the application of external power. Since many variations of power supply circuitry exist, Fairchild cannot guarantee that the power-on clear will operate under every power-up condition.

The power-on clear circuitry contains on-chip sensors to monitor various conditions. The following conditions must be satisfied before the power-reset sequence is allowed to start:

1. Supply voltage must be above a certain value, typically +3 V to +4 V.
2. The clocks of the device must be functioning.
3. The substrate bias must reach a certain level.

All three conditions must be met before the power-on clear circuitry initiates a reset cycle. However, these conditions can be satisfied even with a supply voltage of as low as 3 volts. The latest versions of the F38C70 have a modified delay circuit that gives a typical delay of 500 μs (with a 4 mHz crystal) after the above conditions are met. This is an improvement over the earlier F38C70 versions.

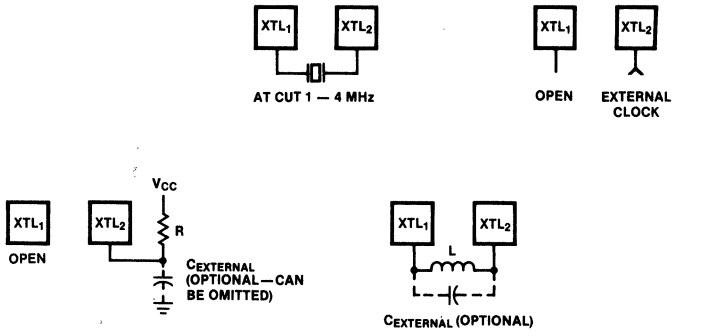
Since the F38C70 is only guaranteed to operate at a supply voltage of 4.5 V or greater, the user must ensure that the supply voltage is at least 4.5 V when the F38C70 initiates the reset cycle. For power supplies having a slow rise time, an external RC network can be converted to the external reset input of the F38C70 to hold the device in a reset state long enough to allow the power supply to reach a voltage of 4.5 V. For example:



External Reset

When the RESET signal is taken low, the contents of the program counter are pushed to the stack register and the program counter and the ICB of the status register are cleared. The original stack register contents are lost. As with power-on clear, ports 4, 5, 6, and 7 are loaded with H'00'. The contents of all other registers and ports are unchanged. When the RESET signal is taken high, the first program instruction is fetched from ROM location H'0000'.

Figure 3 Clock Configurations



Minimum R = 4kΩ

$C = 20.5\ pF \pm 2.5\ pF + C_{EXTERNAL}$

$f_{MIN} \cong \frac{1}{1.1\,RC + 65\ ns}$

$f_{MAX} \cong \frac{1}{1.0\,RC + 15\ ns}$

Example with $C_{EXTERNAL} = 0$

R = 15 kΩ ± 5%

f ≅ 2.9 MHz ± 26%

Minimum L = 0.1 mH

Minimum Q = 40

Maximum $C_{EXTERNAL}$ = 30 pF

f ≅ 3.0 MHz ± 10%

$C = 10\ pF \pm 1.3\ pF + C_{EXTERNAL}$

$f \cong \frac{1}{2\pi\sqrt{LC}}$

Example with $C_{EXTERNAL} = 0$

L = 0.3 mH ± 10%

Test Logic

Special test logic is implemented to allow access to the internal main data bus for test purposes. In normal operation, the TEST pin must be connected to ground. When the TEST signal is set to V_{DD}, port 4 becomes an output of the internal data bus and port 5 becomes a wired-OR input to the internal data bus. The data appearing on the port 4 pins is logically true, whereas input data forced on port 5 must be logically false. When the TEST signal is set to one-half the level of V_{CC} (V_{CC}/2), the ports act as above and the 2K X 8 program ROM is prevented from driving the data bus. In this mode, operands and instructions are forced externally through port 5 instead of being accessed from the program ROM. When the TEST signal is in either the V_{DD}/2 or the high state, the STROBE signal ceases its normal function and becomes a cycle clock (identical to the F8 multi-chip system write clock, except inverted).

The TEST pin capabilities are impractical for user applications because of timing complexities; however these capabilities are sufficient to enable Fairchild to implement rapid methods for thoroughly testing the F38C70.

Clocks

The time bases for the F38C70 originate from one of four external sources by mask options. These four configurations are illustrated in figure 3. External capacitors are not required. In all external clock modes, the external time base frequency is divided by two to form the internal Φ clock. The selection of clock configurations is by mask options.

Figure 4 F38C70 Programmable Registers and Ports

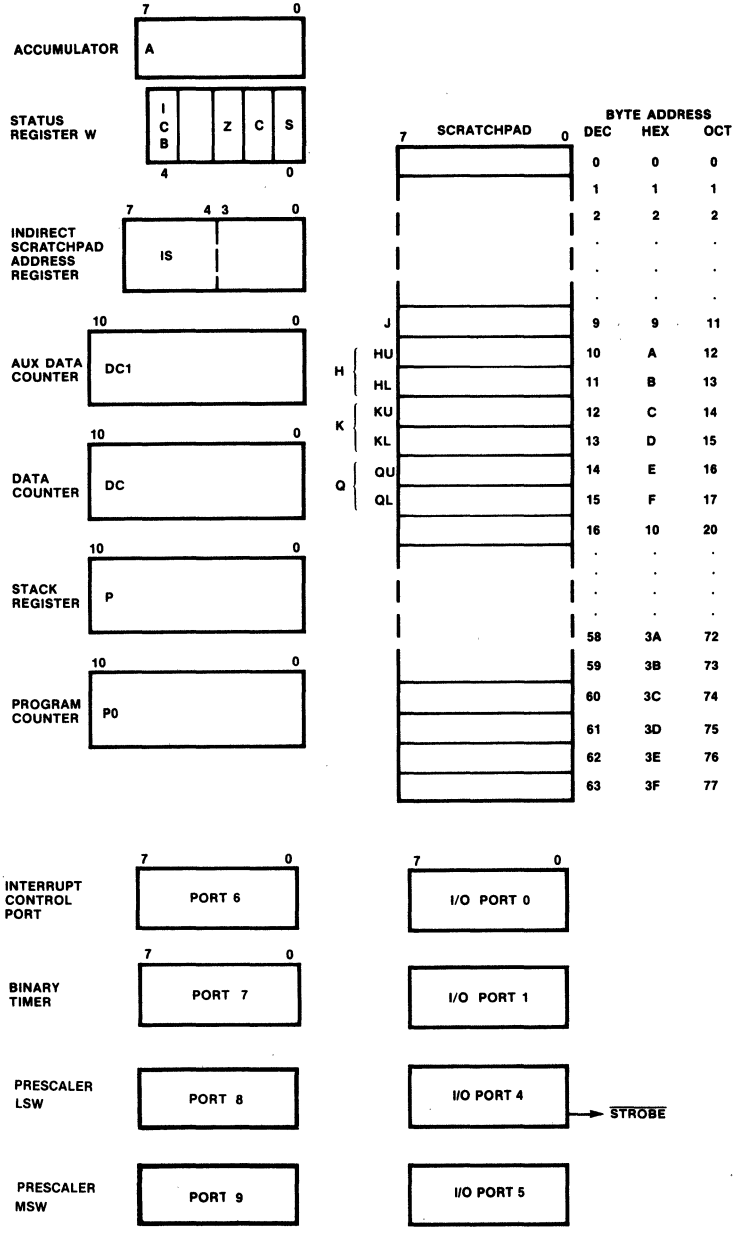


Figure 5 PS Instruction

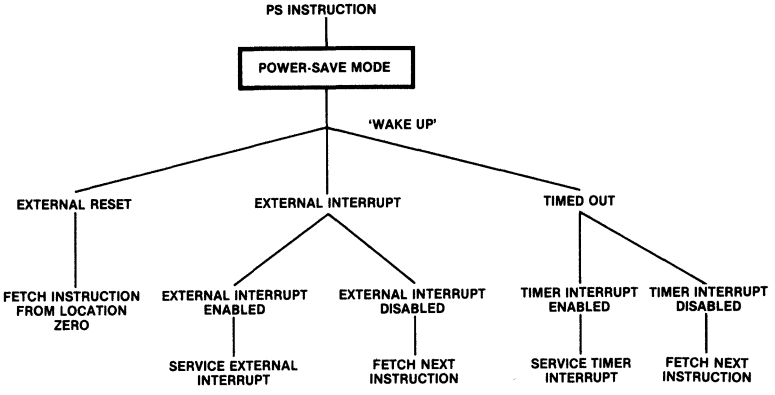
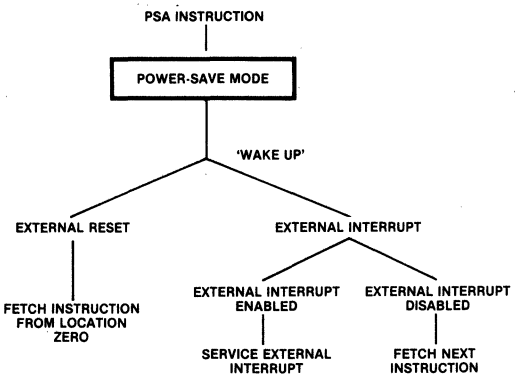


Figure 6 PSA Instruction



Instruction Set

The F38C70 executes the entire instruction set of the F3870 family. In addition, two instructions exclusive to the F38C70 allow the F38C70 to further reduce its power consumption by entering into one of two power-save modes.

A summary of programmable registers and ports is shown in *Figure 4. Table 1* lists the F38C70 instruction set and F8-compatible instructions.

Power-Save Mode

When the power-save instruction (mnemonic PS, OP code 2D) is executed, the F38C70 halts all its operations except the timer and interrupts. The microcomputer is returned to the operating status by an external reset, an external interrupt, or a timer interrupt (as the timer is timed out).

Power-Save All Mode

When the power-save all instruction (mnemonic PSA, Op code 2F) is executed, the F38C70 halts all its operations and goes into a power-save mode (refer to Figures 5 and 6). The microcomputer is returned to the previous operating status by an external reset or an external interrupt. Both the timer and prescaler are reset when PSA is executed, except in the event counter mode.

In returning from either power-save mode, the microcomputer exercises the interrupt routine or continues with the next instruction, depending on whether the interrupt is enabled.

If the return is by an external reset, the microcomputer restarts from the reset mode.

Table 1 F38C70 Instruction Set and F8-Compatible Instructions

Accumulator Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	Zero	CRY	Sign
Add Carry	LNK		ACC ← (ACC) + CRY	19	1	1	1/0	1/0	1/0	1/0
Add Immediate	AI	ii	ACC ← (ACC) H 'ii'	24ii	2	2.5	1/0	1/0	1/0	1/0
And Immediate	NI	ii	ACC → (ACC) H 'ii'	21ii	2	2.5	0	1/0	0	1/0
Clear	CLR		ACC → H'00'	70	1	1				
Compare Immediate	CI	ii	H 'ii'	25ii	2	2.5	1/0	1/0	1/0	1/0
Complement	COM		ACC → (ACC) ⊕H'FF'	18	1	1	0	1/0	0	1/0
Exclusive or Immediate	XI	ii	ACC → (ACC) ⊕ H ii	23 ii	2	2.5	0	1/0	0	1/0
Increment	INC		ACC → (ACC) + 1	1F	1	1	1/0	1/0	1/0	1/0
Load Immediate	LI	ii	ACC → H 'ii'	20 ii	2	2.5	—	—	—	—
Load Immediate Short	LIS	1	ACC → H'0i'	7i	1	1	—	—	—	—
Or Immediate	OI	ii	ACC → (ACC) V H 'ii'	22ii	2	2.5	0	1/0	0	1/0
Shift Left One	SL	1	Shift Left 1	13	1	1	0	1/0	0	1/0
Shift Left Four	SL	4	Shift Left 4	15	1	1	0	1/0	0	1/0
Shift Right One	SR	1	Shift Right 1	12	1	1	0	1/0	0	1/0
Shift Right Four	SR	4	Shift Right 4	14	1	1	0	1/0	0	1/0

Table 1 F38C70 Instruction Set and F8-Compatible Instructions (Continued)

Branch Instructions (In all conditional branches, PO (PO) + 2 if the test conditions are not met. Execution is complete in 30 cycles.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF Zero CRY Sign
Branch on Carry	BC	aa	PO ← [(PO) + 1] + H'aa' if CRY = 1	82 aa	2	3/3.5**	— — — —
Branch on Positive	BP	aa	PO→ [(PO) + 1] + H'aa' if	81aa	2	3/3.5**	— — — —
Branch on Zero	BZ	aa	PO → [(PO) + 1] + H'aa' if Zero = 1	84aa	2	3/3.5**	— — — —
Branch on True	BT	t,aa	PO ← [(PO) + 1] + H'aa' if any test is true	8t aa	2	3/3.5**	— — — —

1 - TEST CONDITION

2^2	2^1	2^0
ZERO	CRY	SIGN

Branch if Negative	BM	aa	PO ← [(PO) + 1] + H'aa' if Sign = 0	91aa	2	3/3.5**	— — — —
Branch if No Carry	BNC	aa	PO ← [(PO) + 1] + H'aa' if Carry ≠ 0	92 aa	2	3/3.5**	— — — —
Branch if No Overflow	BNO	aa	PO ← [(PO)i + 1] + H'aa' if OVF = 0	98 aa	2	3/3.5**	— — — —
Branch if Not Zero	BNZ	aa	PO ← [(PO) + 1] + H'aa' if Zero = 0	94 aa	2	3/3.5**	— — — —
Branch if False Test	BF	t.aa	PO ← [(PO) + 1] + H'aa' if all false test bits	9t aa	2	3/3.5**	— — — —

1 = TEST CONDITION

2^3	2^2	2^1	2^0
OVF	ZERO	CRY	SIGN

Branch If ISAR(Lower)7	BR7	aa	PO← [(PO)+1]+H'aa' if ISARL ≅ 7	8Faa	2	2.5	— — — —
			PO ← (PO) + 2 if ISARL = 7	2.0	— —	— —	
Branch Relative	BR	aa	PO ← [(PO) + 1] + H'aa'	90 aa	2	3.5	— — — —
Jump*	JMP	aaaa	PO ← H'aaa'	29 aaa	3	5.5	— — — —

* Privileged instruction

** 3.5 Cycles if branch taken.

Note
JMP and P1 change accumulator contents to the high byte address.

Table 1 F38C70 Instruction Set and F8-Compatible Instructions (Continued)

Memory Reference Instructions (In all memory reference instructions, the data counter is incremented DC←DC — 1.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF Zero CRY Sign
Add Binary	AM		ACC←(ACC) + [(DC)]	88	1	2.5	1/0 1/0 1/0 1/0
Add Decimal	AMD		ACC←(ACC) + [(DC)]	89	1	2.5	1/0 1/0 1/0 1/0
AND	NM		ACC←(ACC)∧[(DC)]	8A	1	2.5	0 1/0 0 1/0
COMPARE	CM		[(DC)] + (ACC) + 1	8D	1	2.5	1/0 1/0 1/0 1/0
EXCLUSIVE OR	XM		ACC←(ACC) ⊕ [(DC)]	8C	1	2.5	0 1/0 0 1/0
LOAD	LM		ACC←[(DC)]	16	1	2.5	— — — —
LOGICAL OR	OM		ACC←(ACC) ∨[(DC)]	8B	1	2.5	0 1/0 0 1/0
STORE	ST		(DC)←(ACC)	17	1	2.5	— — — —

Address Register Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF Zero CRY Sign
Add to Data Counter	ADC		DC←(DC) + (ACC)	8E	1	2.5	— — — —
Call to Subroutine	PK		P←(PO)ΦPOU←(r12) + PL←(r13)	DC	1	4	— — — —
Call to Subroutine Immediate	PI	aaaa	P←(P)PO←H'aaaa' ↕	28aaaa	3	6.5	— — — —
Exchange DC	XDC		DC↕DC1	2C	1	2	— — — —
Load Data Counter	LR	DC,Q	DCU←(r14), DCL←(r15)	0F	1	4	— — — —
Load Data Counter	LR	DC,H	DCU←(r10), DCL←(r11)	10	1	4	— — — —
Load DC Immediate	DCI	aaaa	DC←H'aaaa'	2Aaaaa	3	6	— — — —
Load Program Counter	LR	PO,Q	POU←(r14), POL←(r15)	0D	1	4	— — — —
Load Stack Register	LR	P,K	PU←(r12), PL←(r13)	09	1	4	— — — —
Return From Subroutine	POP		PO\P	1C	1	2	— — — —
Store Data Counter	LR	Q,DC	r14←(DCU), r15←(DCL)	0E	1	4	— — — —
Store Data Counter	LR	H,DC	r10←(DCU), r11←(DCL)	11	1	4	— — — —
Store Stack Register	LR	K,P	r12←(PU), r13←P	08	1	4	— — — —

Table 1 F38C70 Instruction Set and F8-Compatible Instructions (Continued)

Scratchpad Register Instructions (refer to scratchpad addressing modes.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	Zero	CRY	Sign
Add Binary	AS	r	ACC(ACC) + (r)	Cr	1	1	1/0	1/0	1/0	1/0
Add Decimal	ASD	r	ACC←(ACC) + (r)	Dr	1	2	1/0	1/0	1/0	1/0
Decrement	DS	r	r←(r) + H'FF'	3r	1	1.5	1/0	1/0	1/0	1/0
Load	LR	A,r	ACC←(r)	4r	1	1	—	—	—	—
Load	LR	A,KU	AC←(r12)	00	1	1	—	—	—	—
Load	LR	A,KL	ACC←(r13)	01	1	1	—	—	—	—
Load	LR	A,QU	ACC←(r14)	02	1	1	—	—	—	—
Load	LR	A,QL	ACC←(r15)	03	1	1	—	—	—	—
Load	LR	r,A	r←(ACC)	5r	1	1	—	—	—	—
Load	LR	KU,A	r12←(ACC)	04	1	1	—	—	—	—
Load	LR	KL,A	r13←(ACC)	05	1	1	—	—	—	—
Load	LR	QU,A	r14←(ACC)	06	1	1	—	—	—	—
Load	LR	QL,A	r15←(ACC)	07	1	1	—	—	—	—
And	NS	r	ACC←(ACC)∧(r)	Fr	1	1	0	1/0	0	1/0
Exclusive Or	XS	r	ACC←(ACC)⊕(r)	Er	1	1	0	1/0	0	1/0

Table 1 F38C70 Instruction Set and F8-Compatible Instructions (Continued)

Miscellaneous Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	Zero	CRY	Sign
Disable Interrupt	DI		Reset ICB	1A	1	2	—	—	—	—
Enable Interrupt*	EI		SET ICB	1B	1	2	—	—	—	—
Input	IN	aa	ACC ← (Input PORT aa)	26aa	2	4	0	1/0	0	1/0
Input Short	INS	a	ACC ← (Input PORT a)	Aa	1	4***	0	1/0	0	1/0
Load ISAR	LR	IS.A	ISAR ← (ACC)	0B	1	1	—	—	—	—
Load ISAR Lower	LISL	a	ISARL ← a	01101a**	1	1	—	—	—	—
Load ISAR Upper	LISU	a	ISARU ← a	01100**	1	1	—	—	—	—
Load statusregister	LR	W.J	W ← (r9)	1D	1	2	1/0	1/0	1/0	1/0
No-Operation	Nop		P0 ← (P0) + 1	2B	1	1	—	—	—	—
OUTPUT	OUT	aa	OUTPUT PORT aa ← (ACC)	27 aa	2	4	—	—	—	—
OUTPUT Short	OUTS	a	OUTPUT PORT a ← (ACC)	Ba	1	4***	—	—	—	—
Store ISAR	LR	A.15	ACC ← (ISAR)	0A	1	1	—	—	—	—
Store Status Reg	LR	J.W	r9 ← (W)	1E	1	1	—	—	—	—
Power Save	PS		Halt Internal Clock	2D	1	3	—	—	—	—
Power Save All	PSA		Halt Internal Clock and Timer	2F	1	3	—	—	—	—

*Privileged instruction
**3-bit octal digit
***Two machine cycles for CPU ports

Notes
Each lower case character represents a hexadecimal digit.
Each cycle equals four machine clock periods.
Lower case denotes variables specified by programmer.

Function definitions

←	is replaced by
( )	the contents of
(-)	binary ones complement of
+	arithmetic add (binary or decimal)
⊕	logical OR exclusive
	logical AND
	logical OR inclusive
H'#	hexadecimal digit

Register Names

a	address variable
	A accumulator
DC	data counter (indirect address register)
DCI	data counter #1 (auxiliary data counter)
DCL	least significant eight bits of data counter addressed
DCU	most significant eight bits of data counter addressed
H	scratchpad register #10 and #11
i and ii	immediate operand
ICB	interrupt control bit
IS	indirect scratchpad address register
ISAR	indirect scratchpad address register
ISARL	least significant three bits of ISAR
ISARU	most significant three bits of ISAR
J	scratchpad register #
K	registers #12 and #13
KL	register #13
KU	register #12
P0	program counter
P0L	least significant eight bits of program counter
P0U	most significant eight bits of program counter
P	stack register
PL	least significant eight bits of program counter
PU	most significant eight bits of active stack register
Q	registers #14 and #15
QL	register #15
QU	register #14
r	scratchpad/register (any address through 11)
W	status register

Scratchpad Addressing Modes (Machine Code Format)

r = C	(hexadecimal) register addressed by ISAR (unmodified)
r = D	(hesadecimal) register addressed by ISAR, ISARL incremented
r = E	(hexadecimal) Register addressed by ISAR, ISARL decremented
r = F	(no operation performed)
r = 0-B	(hexadecimal) register 0 through 11 addressed directly from the instruction

Status Register

-	no change in condition
1/10	is set to 1 or 0 depending on conditions
CRY	carry flag

Supplementary Notes

The interrupt control bit of the status register is automatically reset when an interrupt request is acknowledged. It is then the programmer's responsibility to determine when ICB will again be set (by executing an EI instruction). This action prevents an interrupt service routine from being interrupted, unless the programmer so desires.

When reading the interrupt control port (port 6), bit 7 of the accumulator 1, loaded with the actual logic level being applied to the EXT INT pin, regardless of the status of ICP bit 2 (the EXT INT active level bit); that is, if EXT INT is at +5 V, bit 7 of the accumulator is set to a logic 1, but if EXT INT is at V_{SS}, the accumulator bit 7 is reset to logic 0.

In the instruction set summary (table 1), the number of cycles shown are nominal machine cycles. A nominal cycle is defined as 4 Φ clock periods, thus requiring 2 μs for a 2-mHz clock frequency (4-mHz external time base frequency). When desired, the long machine cycles can be altered to short machine cycles by mask option.

The following nomenclature for register names is used for consistency with the assembly language mnemonics:

F8	F38C70	Register
PC_0	PO	program counter
PC_1	P	stack register
DC_0	DC	data counter
DC_1	DC1	auxiliary data counter

For the F38C70, execution of an INS or OUTS instruction requires two machine cycles for ports 0 and 1, whereas ports 4 an 5 require four machine cycles. When an external reset of the F38C70 occurs, PO is stored in P and the old contents of P are lost. Note that an external reset is recognized at the start of the machine cycle and not necessarily at the end of an instruction. Thus, if the F38C70 is executing a multi-cycle instruction, that instruction is not completed, and the contents of P, upon reset, may not necessarily be the address of the instruction that would have been executed next. They may, for example, point to an immediate operand, if the reset occurred during the second cycle of an LI or CI instruction. Additionally, several instructions (JMP, PI, PK, LR, PO, Q) as well as the interrupt acknowledge sequence, modify PO in parts. That is, they alter PO by first loading one part, then the other part, and the entire operation takes more than one cycle. Should reset occur during this modification process, the value stored in P becomes part of the old PO (the not yet modified part), and part of the new PO (already modified part). Thus, care should be taken (perhaps by external gating) to ensure that reset does not occur at an undesirable time, if any significance is to be given to the contents of P after a reset occurs.

If desired, the F38C70 can execute all instructions in short cycles via mask options to improve the execution speed of the device.

Signal Descriptions

The F38C70 input and output signals are described in Table 2.

Table 2 F38C70 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
XTL_1 XTL_2	1	Clock	The time base inputs to which a crystal (1 to 4 mHz), LC network, RC network, or an external single-phase clock is connected.
I/O Ports			
$\overline{PO}_0 - \overline{PO}_7$	3, 4, 5, 6, 19, 18 17, 16	Port Address	The 32 ports are individually used as either TTL-compatible inputs or as latched outputs.
$\overline{P1}_0 - \overline{PO}_7$	37, 36, 35, 34, 22, 23, 24, 25	Port Address	
$\overline{P4}_0 - \overline{P4}_7$	8, 9, 10, 11, 12, 13, 14, 15,	I/O Port	
$\overline{P5}_0 - \overline{P5}_7$	33, 32, 31, 30, 29, 28, 27, 26	I/O Port	
Interrupt/Reset			
EXT INT	38	External Interrupt	The active state of the external interrupt signal is software programmable; it is also used in conjunction with the timer for pulse width measurement and event counting.
RESET	39	Reset	This input signal is used to reset the F38C70 externally. When the signal is allowed to go low, the F38C70 resets. When subsequently allowed to go high, the F38C70 begins program execution at location H'0000'.
Strobe			
$\overline{STROBE}$	7	Strobe	This output pin, which is normally high, provides a single low pulse after valid data is present on the $P4_0 - P4_7$ pins during an output instruction.
Test			
TEST	21	Test	An input signal used only in testing the F38C70. For normal circuit function, this pin must be connected to ground.
Power			
V_{SS}	20	Ground	Common power and signal return
V_{CC}	40	Power Supply	Power supply input signal, +5 (± 10%) V

DC Characteristics

The characteristics of the F38C70 are provided in table 3.

Table 3 F38C70 DC Charateristics $T_A = 0°$ to 70 °C, $V_{CC} = 5V \pm 10\%$, I/O Power Dissipation ≤ mW

Symbol	Parameter	Min	Max	Unit	Test Conditions
I_{CC}	Power Supply Current		TBD	mA	Outputs Open
P_D	Power Dissipation		TBD	mW	Outputs Open
V_{IHEX}	External Clock Input High Voltage	2.4	5.8	V	
V_{ILHEX}	External Clock Input Low Voltage	−0.3	0.6	V	
I_{HEX}	External Clock Input High Current		100	μA	V_{IHEX} = 2.4 V
I_{LEX}	External Clock Input Low Current		−100	μA	V_{ILEX} = 0.6 V
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.3	V	
V_{IL}	Input Low Voltage	−0.3	0.8	V	
I_{IH}	Input High Current (except 3-state option)		100	μA	V_{IH} = 2.4 V, internal pull-up
I_{IL}	Input Low Current (except open drain and direct drive ports)		−1.6	mA	V_{IL} = 0.4 V
I_{LOD}	Leakage Current		±10	μA	$0 \leq V_{IN} \leq V_{CC}$
I_{OH}	Output High Current (except open drain and direct drive ports) std.	−100		μA	V_{OH} = 2.4 V
I_{OHDD}	Output Drive Current (push-pull)	−1.5	TBD	mA	V_{OH} = 0.7 V to 1.5 V
I_{OL}	Output Low Current	1.8		mA	V_{OL} = 0.4 V
I_{OHS}	Output High Current (STROBE Output)	−300		μA	V_{OH} = 2.4 V
I_{OLS}	Output Low Current (STROBE Output)	5.0		mA	V_{OL} = 0.4 V

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Temperature (Ambient) Under Bias	0°C, +70°C
Storage Temperature	−55°C, +150°C
Voltage on Any Pin with Respect to Ground (Except Open Drain Pins)	−0.3 V, V_{CC} + 0.3 V
Power Dissipation	1 W

Ordering Information

Part Number	Temperature Package	Range*
F38C70DC	Ceramic	C
F38C70DL	Ceramic	L
F38C70DM	Ceramic	M
F38C70PC	Plastic	C

*C = Commercial Temperature Range 0°C to +70°C
L = Limited Temperature Range −40°C to +85°C
M = Military Temperature Range −55°C to + 125°C

FAIRCHILD
A Schlumberger Company

F38E70
Single-Chip Microcomputer

Microprocessor Product

Description

The Fairchild single-chip microcomputer series offers a variety of circuits for the high-volume, cost-sensitive markets. The F38E70 is a complete 8-bit microcomputer on a single MOS integrated circuit. The F38E70 is functionally identical to the F3870, except the F38E70 has 2K bytes of EPROM in place of 2K bytes of ROM. The F38E70 can execute the F8 instruction set of more than 70 commands. The device features 2048 bytes of EPROM, 64 bytes of scratchpad RAM, a programmable binary timer, 32 bits of I/O, and a single +5 V power supply requirement.

Utilizing Fairchild's double-ion-implant, N-channel technology and advanced circuit design techniques, the single-chip F38E70 offers maximum cost-effectiveness in many low-to-medium volume systems. When production volume requires large quantities, the transition to the mask-programmed F3870 is very straightforward, with no circuit design changes.

- **Single-Chip Microcomputer**
- **Software-Compatible with F8 Family**
- **2048-Byte EPROM (F38E70-2)**
- **64-Byte Scratchpad RAM**
- **32-Bit (4-Port) TTL-Compatible I/O**
- **Programmable Binary Timer**
  - **Interval Timer Mode**
  - **Pulse Width Measurement Mode**
  - **Event Counter Mode**
- **External Interrupt**
- **Crystal, LC, RC, External, or Internal Time Base**
- **Low Power (375 mW Typical)**
- **Single +5 V ±10% Power Supply**
- **Simple EPROM Programming**

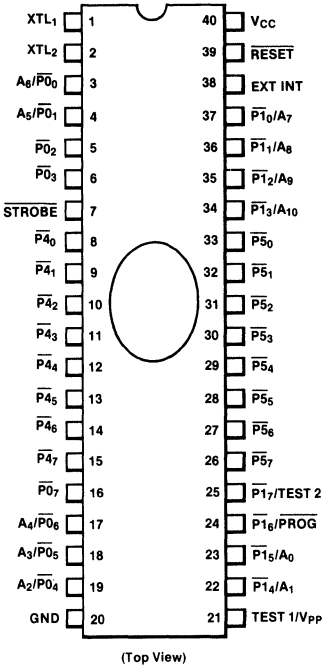
Pin Names

$\overline{P0}_0$-$\overline{P0}_7$	Bidirectional I/O Port 0/Address*
$\overline{P1}_0$-$\overline{P1}_7$	Bidirectional I/O Port 1/Address*
$\overline{P4}_0$-$\overline{P4}_7$	Bidirectional I/O Port 4/Data Out*
$\overline{P5}_0$-$\overline{P5}_7$	Bidirectional I/O Port 5/Data In*
$\overline{STROBE}$	Ready Strobe Output
EXT INT	External Interrupt Input
$\overline{RESET}$	External Reset Input
TEST 1/V_{PP}	Test Line/PROG Voltage Input**
XTL_1, XTL_2	Time Base Input
V_{CC}, GND	Power Supply Lines

*As shown in the connection diagram, some port 0 and port 1 pins are address inputs for programming the F38E70 EPROM section. Ports 4 and 5, Data Out and In, refer to the programming and test modes.

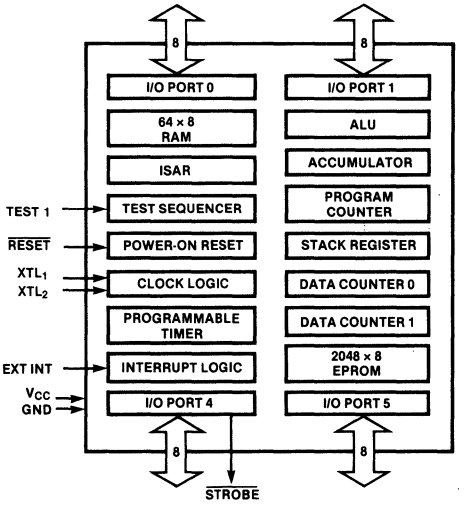
**Caution: applying +25 V to the V_{PP} pin without the presence of V_{CC} will damage the device.

Connection Diagram
40-Pin DIP



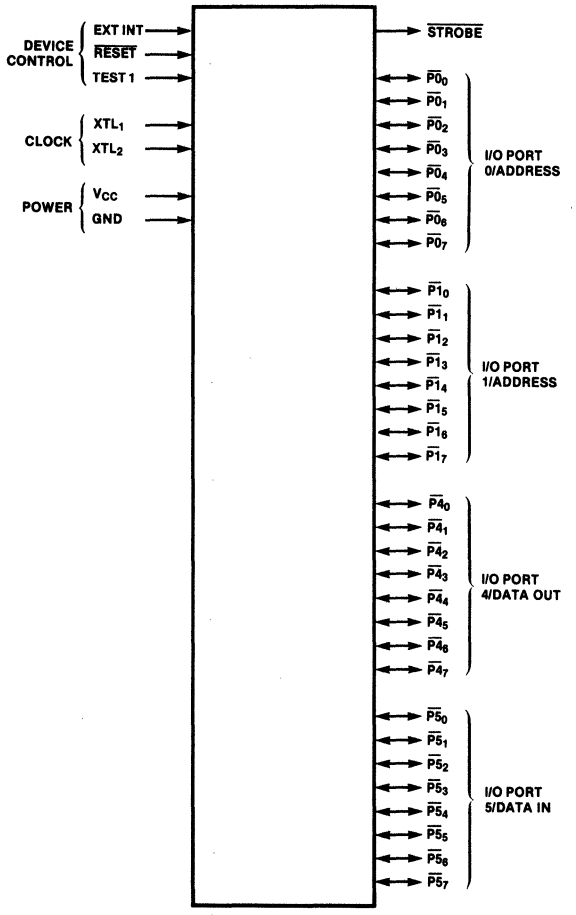
(Top View)

F38E70 Architecture



4

Signal Functions



Device Organization

This section describes the basic functional elements of the F38E70 as shown in *Figure 1*.

Main Control Logic

The instruction register (IR) receives the operation code (OP code) of the instruction to be executed from the program EPROM via the data bus. During all OP code fetches, eight bits are latched into the IR. Some instructions are completely specified by the upper four bits of the OP code. In those instructions, the lower four bits are an immediate register address or an immediate 4-bit operand. Once latched into the IR, the main control logic decodes the instruction and provides the necessary control gating signals to all circuit elements.

EPROM Address Registers

There are four 11-bit registers associated with the 2K x 8 EPROM. These are the program counter (P0), the stack register (P), the data counter (DC), and the auxiliary data counter (DC1). The program counter is used to address instructions or immediate operands. The stack register is used to save the contents of P0 during an interrupt or subroutine cell. Thus, P contains the return address at which processing is to resume upon completion of the subroutine or the interrupt routine.

The data counter (DC) is used to address data tables. This register is auto-incrementing. Of the two data counters, only DC can access the EPROM. However, the XDC instruction allows DC and DC1 to be exchanged.

Associated with the address registers is an 11-bit adder/incrementer. This logic element is used to increment P0 or DC when required, and is also used to add displacements to P0 on relative branches or to add the accumulator contents to DC1 with the ADC (add data counter) instruction.

2048 x 8 EPROM

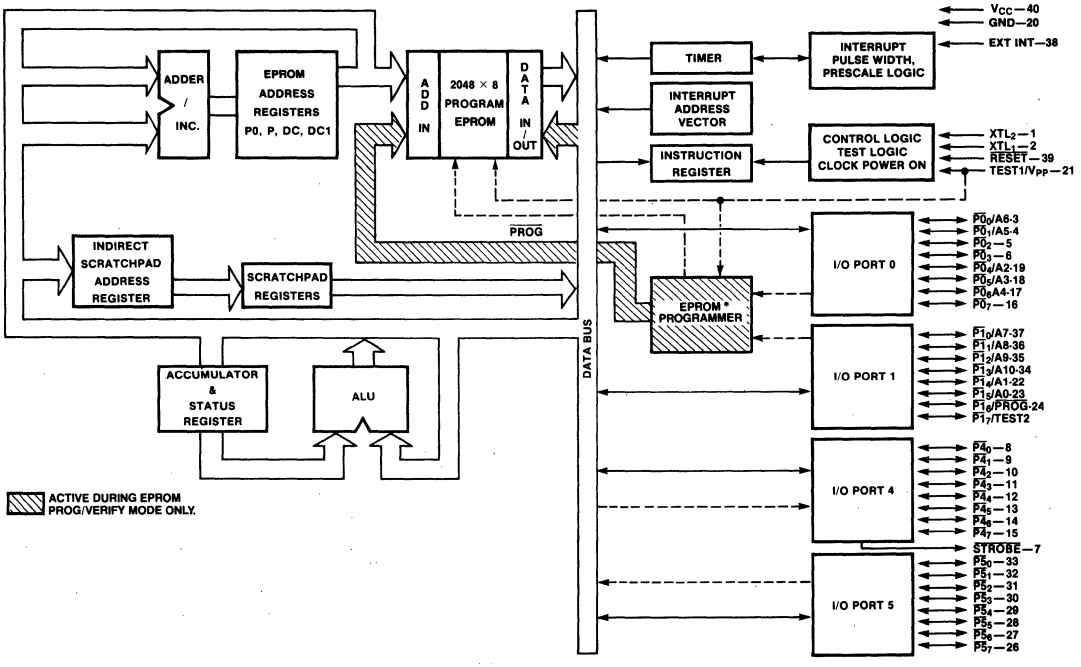
The microcomputer program and data constants are stored in the program EPROM. When an EPROM access is required, the appropriate address register (P0 or DC) is gated onto the EPROM address bus and the EPROM output is gated onto the main data bus. The first byte in the EPROM is location zero.

Scratchpad and ISAR

The scratchpad provides 64 8-bit registers that may be used as general-purpose read/write data memory. The indirect scratchpad address register (ISAR) is a 6-bit register used to address the 64 registers. All 64 registers may be accessed using ISAR. In addition, the lower order 12 registers may also be directly addressed.

The ISAR can be visualized as holding two octal digits. This division of ISAR is important, since a number of instructions increment or decrement only the least significant three bits of ISAR when referencing scratchpad bytes via ISAR. This makes it easy to reference a buffer consisting of up to eight contiguous scratchpad bytes. For example, when the low-order octal digit is incremented or decremented, ISAR is incremented from 27_8 to 20_8 or is decremented from 20_8 to 27_8. This feature of the ISAR is very useful in many

Fig. 1 Block Diagram



Pin Functions

Pin Name	Type	Description
$\overline{P0}_0$-$\overline{P0}_7$ $\overline{P1}_0$-$\overline{P1}_7$ $\overline{P4}_0$-$\overline{P4}_7$ $\overline{P5}_0$-$\overline{P5}_7$	Input/Output	Thirty-two lines that can be individually used as either TTL-compatible inputs or as latched outputs. For EPROM programming, 11 lines of ports 0 and 1 are used as address inputs and one line of port 1 is a program control. Port 5 is EPROM data input, and port 4 is EPROM output for verification.
$\overline{STROBE}$	Output	This pin, which is normally HIGH, provides a single LOW pulse after valid data is present on the $\overline{P4}_0$-$\overline{P4}_7$ pins during an output instruction.
$\overline{RESET}$	Input	$\overline{RESET}$ may be used to externally reset the F38E70. When pulled LOW, the F38E70 resets. When then allowed to go HIGH, the F38E70 begins program execution at the program location H '0000'. RESET is held LOW during EPROM programming.
EXT INT	Input	The external interrupt input. Its active state is software-programmable. This input is also used in conjunction with the timer for pulse width measurement and event counting.
XTL_1, XTL_2	Input	The time base inputs to which a crystal (1 to 4 MHz), LC network, RC network, or an external single-phase clock may be connected. If timing is not critical, the F38E70 operates from its internal oscillator with no external components.
TEST 1/V_{PP}	Input	An input used only in testing and programming the F38E70. For normal circuit functionality, this pin is left unconnected or may be grounded. For EPROM programming, the test pin is connected to the programming voltage (typically 23 V).
$P1_7$/TEST 2	Input	I/O during normal operation; must be HIGH when in verify mode.
V_{CC}	Power	V_{CC} is the power supply input (+5 V ± 10%).

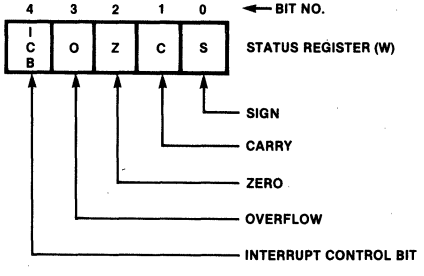
program sequences. All six bits of ISAR may be loaded at one time, or either half may be loaded independently.

Scratchpad registers 9 through 15 (decimal) are given mnemonic names (J, H, K, and Q) because of special linkages between these registers and other registers, such as a stack register. These special linkages facilitate the implementation of multi-level interrupts and subroutine nesting. For example, the instruction LR K, P stores the lower eight bits of the stack register into register 13 (K lower, or KL) and stores the upper three bits of P into the three least significant bits of register 12 (K upper, or KU).

Arithmetic and Logic Unit (ALU)
After receiving commands from the main control logic, the ALU performs the required arithmetic or logic operations (using the data presented on the two input busses) and provides the result on the result bus. The arithmetic operations that can be performed in the ALU are binary add, decimal adjust, add with carry, decrement, and increment. The logic operations that can be performed are AND, OR, exclusive-OR, ones complement, shift right, and shift left. Besides providing the result on the result bus, the ALU also provides four signals representing the status of the result. These signals, stored in the status register (W), represent the CARRY, OVERFLOW, SIGN, and ZERO condition of the result of the operation.

Accumulator
The accumulator (ACC) is the principal register for data manipulation within the F38E70. The ACC serves as one input to the ALU for arithmetic or logical operations. The results of ALU operations are stored back into the ACC.

Status Register
The status register (W) holds five status flags, as follows:



Summary of Status Bits

$\text{OVERFLOW} = \text{CARRY}_7 \oplus \text{CARRY}_6$

$\text{ZERO} = \overline{\text{ALU}_7} \wedge \overline{\text{ALU}_6} \wedge \overline{\text{ALU}_5} \wedge \overline{\text{ALU}_4} \wedge \overline{\text{ALU}_3} \wedge \overline{\text{ALU}_2} \wedge \overline{\text{ALU}_1} \wedge \overline{\text{ALU}_0}$

$\text{CARRY} = \text{CARRY}_7$

$\text{SIGN} = \overline{\text{ALU}_7}$

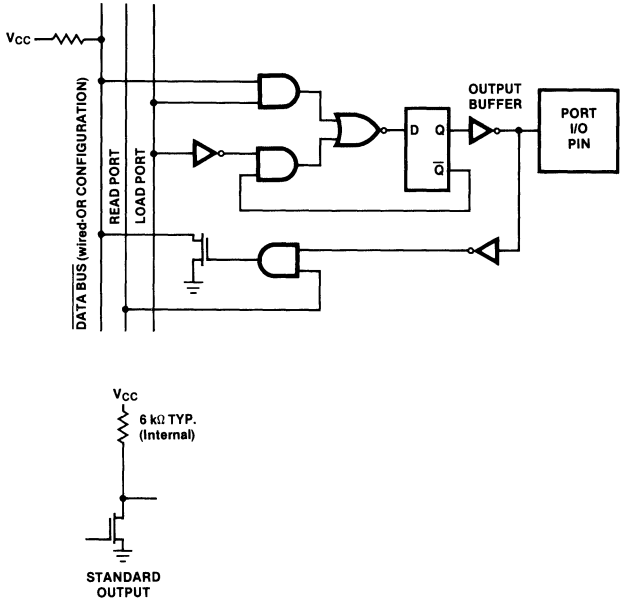
Interrupt Control Bit—The ICB may be used to allow or disallow interrupts in the F38E70. This bit is not the same as the two interrupt enable bits in the interrupt control port (ICP). If the ICB is set and the F38E70 interrupt logic communicates an interrupt request to the CPU section, the interrupt is acknowledged and processed upon completion of the first non-privileged instruction. If the ICB is cleared, an interrupt request is not acknowledged or processed until the ICB is set again.

I/O Ports
The F38E70 provides four complete bidirectional input/output ports: these are ports 0, 1, 4, and 5. An output instruction (OUT or OUTS) causes the contents of the ACC to be latched into the addressed port. An input instruction (IN or INS) transfers the contents of the port to the ACC (port 6 is an exception, which is described later). The I/O buffers on the F38E70 are logically inverted. The schematic of an I/O port is shown in *Figure 2*.

An output ready strobe is associated with port 4. This flag may be used to signal a peripheral device that the F38E70 has just completed an output of new data to port 4. The strobe provides a single low pulse shortly after the output operation is completed, so either edge may be used to signal the peripheral. The $\overline{\text{STROBE}}$ signal may also be used to request new input information from a peripheral simply by doing a dummy output of H '00' to port 4 after completing the input operation.

Timer and Interrupt Control Port
The timer is an 8-bit binary down counter that is software-programmable to operate in one of three modes: the interval timer mode, the pulse width measurement mode, or the event counter mode (the timer characteristics are described in *Table 1*). As shown in *Figure 3*, associated with the timer are an 8-bit register called the interrupt control port, a programmable prescaler, and an 8-bit modulo-N register; *Figure 4* illustrates the timer/interrupt function.

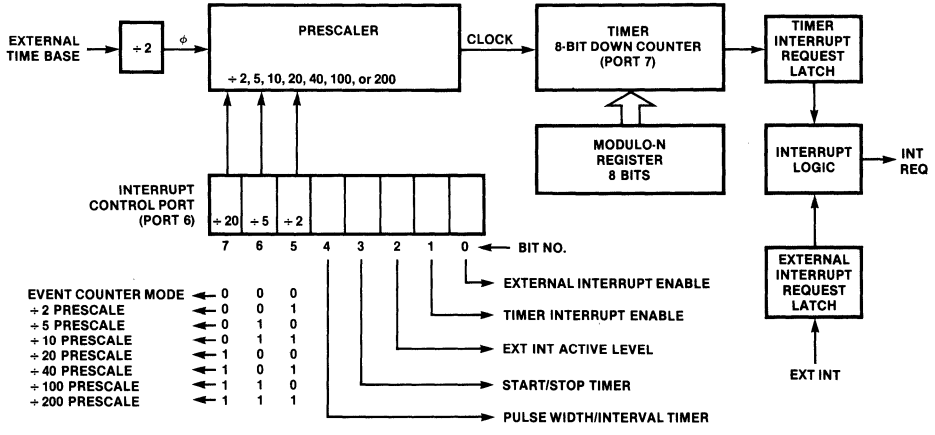
Fig. 2 I/O Port Diagram



All ports are standard output type only.

The $\overline{\text{STROBE}}$ output is always configured similar to a standard output, except that it is capable of driving three TTL loads.

Fig. 3 Timer and Interrupt Control Port Block Diagram



See *Figure 4* for a more detailed functional diagram.

Table 1 Timer Characteristics

Definitions	
Error = indicated time value – actual time value	
tpsc = tφ × prescale value	
Interval Timer Mode	
Single interval error, free-running (note 3)	± 6tφ
Cumulative interval error, free-running (note 3)	0
Error between two timer reads (note 2)	± (tpsc + tφ)
Start timer to stop timer error (notes 1, 4)	+ tφ to – (tpsc + tφ)
Start timer to read timer error (notes 1, 2)	– 5tφ to – (tpsc + 7tφ)
Start timer to interrupt request error (notes 1, 3)	– 2tφ to – 8tφ
Load timer to stop timer error (note 1)	+ tφ to – (tpsc + 2tφ)
Load timer to read timer error (notes 1, 2)	– 5tφ to – (tpsc + 8tφ)
Load timer to interrupt request error (notes 1, 3)	– 2tφ to – 9tφ
Pulse Width Measurement Mode	
Measurement accuracy (note 4)	+ tφ to – (tpsc + 2tφ)
Minimum pulse width of EXT INT pin	2tφ
Event Counter Mode	
Minimum active time of EXT INT pin	2tφ
Minimum inactive time of EXT INT pin	2tφ

Notes

1. All times that entail loading, starting, or stopping the timer are referenced from the end of the last machine cycle of the OUT or OUTS instruction.
2. All times that entail reading the timer are referenced from the end of the last machine cycle of the IN or INS instruction.
3. All times that entail the generation of an interrupt request are referenced from the start of the machine cycle in which the appropriate interrupt request latch is set. Additional time may elapse if the interrupt request occurs during a privileged or multi-cycle instruction.
4. Error may be cumulative if operation is repetitively performed.

The desired timer mode, prescale value, starting and stopping the timer, active level of EXT INT pin, and local enabling or disabling of interrupts are selected by outputting the proper bit configuration from the accumulator to the interrupt control port (port 6) with an OUT or OUTS instruction. Bits within the interrupt control port are defined as follows:

Interrupt Control Port (Port 6)
Bit 0 — External Interrupt Enable
Bit 1 — Timer Interrupt Enable
Bit 2 — EXT INT Active Level
Bit 3 — Start/Stop Timer
Bit 4 — Pulse Width/Interval Timer
Bit 5 — ÷ 2 Timer Prescale Values
Bit 6 — ÷ 5 Timer Prescale Values
Bit 7 — ÷ 20 Timer Prescale Values

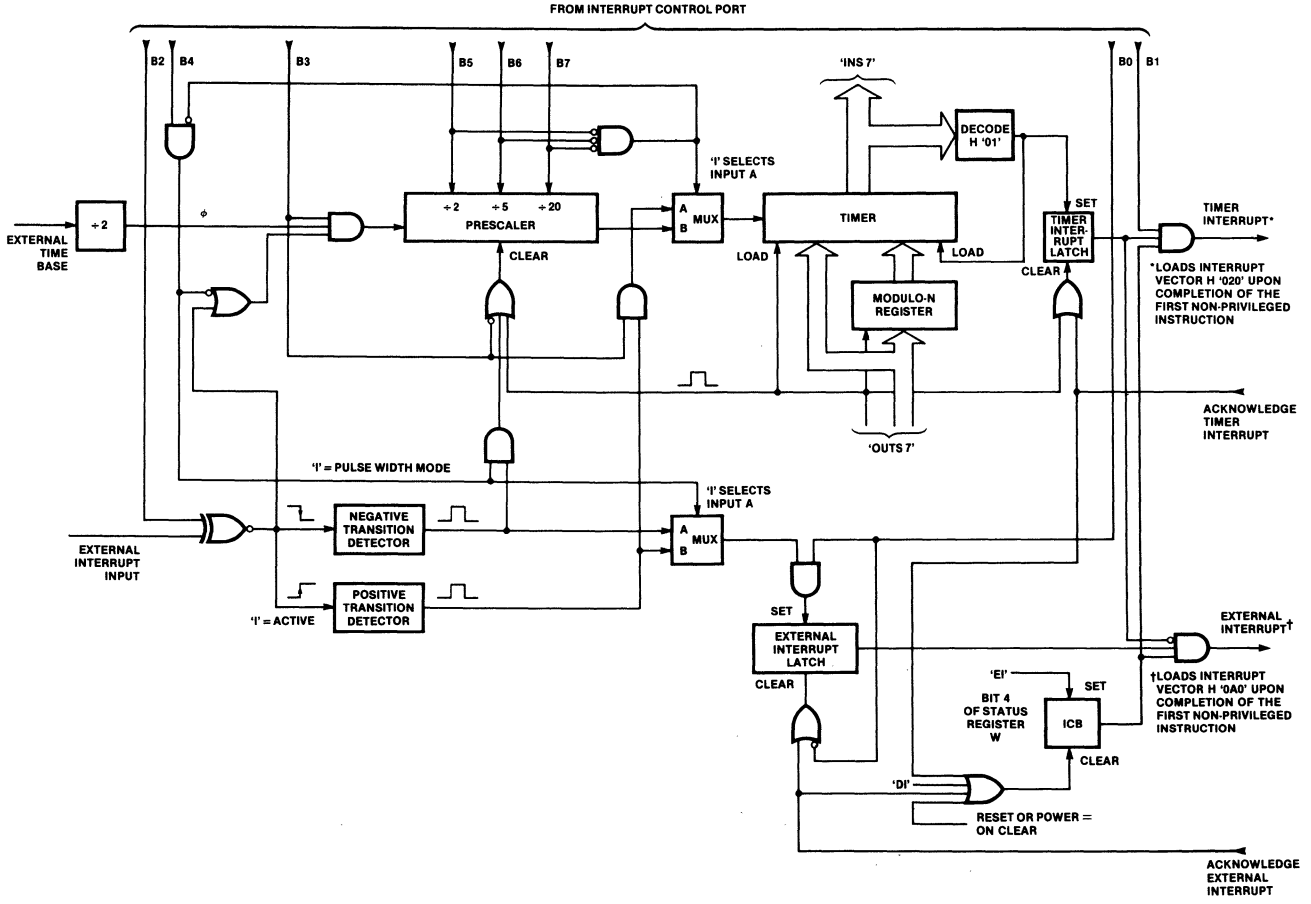
A special situation exists when reading the interrupt control port (with an IN or INS instruction). The accumulator is **not** loaded with the content of the ICP; instead, accumulator bits 0 through 6 are loaded with 0s, while bit 7 is loaded with the logic level being applied to the EXT INT pin, thus allowing the status of EXT INT to be determined without the necessity of servicing an external interrupt request. This capability is useful in establishing a high-speed polled handshake procedure or for using EXT INT as an extra input pin if external interrupts are not required and the timer is used only in the interval timer mode.

The rate at which the timer is clocked in the interval timer mode is determined by the frequency of an internal φ clock and by the division value selected for the prescaler. (The internal φ clock operates at one-half the external time base frequency.) If ICP bit 5 is set and bits 6 and 7 are cleared, the prescaler divides φ by two. Likewise, if bit 6 or 7 is individually set, the prescaler divides φ by 5 or 20, respectively. Combinations of bits 5, 6, and 7 may also be selected. For example, if bits 5 and 7 are set while 6 is cleared, the prescaler divides by 40. Thus, possible prescaler values are ÷ 2, ÷ 5, ÷ 10, ÷ 20, ÷ 40, ÷ 100, and ÷ 200.

Any of three conditions causes the prescaler to be reset: when the timer is stopped by clearing the ICP bit 3, on execution of an output instruction to port 7 (the timer is assigned port address 7), or on the trailing edge transition of the EXT INT pin when in the pulse width measurement mode. These last two conditions are explained in more detail below.

An OUT or OUTS instruction to port 7 loads the content of the accumulator to both the timer and the 8-bit modulo-N register, resets the prescaler, and clears any previously stored timer interrupt request. As previously noted, the timer is an 8-bit down counter that is clocked by the prescaler in the interval timer mode and in the pulse width measurement mode. The prescaler is not used in the event counter mode. The modulo-N register is a buffer whose function is to save the value that was most recently output to port 7. The modulo-N register is used in all three timer modes.

Fig. 4 Timer/Interrupt Functional Diagram



Interval Timer Mode—When ICP bit 4 is cleared (logic 0) and at least one prescale bit is set, the timer operates in the interval timer mode. When bit 3 of the ICP is set, the timer starts counting down from the modulo-N value. After counting down to H '01', the timer returns to the modulo-N value at the next count. On the transition from H '01' to H 'N', the timer sets a timer interrupt request latch. Note that the interrupt request latch is set by the transition of H 'N' in the timer, thus allowing a full 256 counts if the modulo-N register is preset to H '00'. If bit 1 of the ICP is set, the interrupt request is passed on to the CPU section of the F38E70. However, if bit 1 of the ICP is a logic 0, the interrupt request is not passed on to the CPU section but the interrupt request latch remains set. If ICP bit 1 is subsequently set, the interrupt request is then passed on to the CPU section. (Recall from the discussion of the status register interrupt control bit that the interrupt request is acknowledged by the CPU section only if ICB is set.) Only two events can reset the timer interrupt request latch: when the timer interrupt request is acknowledged by the CPU section, or when a new load of the modulo-N register is performed.

Consider an example in which the modulo-N register is loaded with H '64' (decimal 100). The timer interrupt request latch is set at the 100th count following the timer start, and the timer interrupt request latch is repeatedly set on precise 100-count intervals. If the prescaler is set at $\div 40$, the timer interrupt request latch is set every 4000 ϕ clock periods. For a 2 MHz ϕ clock (4 MHz time base frequency), this produces 2 ms intervals.

The range of possible intervals is from 2 to 51,200 ϕ clock periods (1 μs to 25.6 ms for a 2 MHz ϕ clock). However, approximately 50 ϕ periods is a practical minimum because the time between setting the interrupt request latch and the execution of the first instruction of the interrupt service routine is at least 29 ϕ periods (the response time is dependent upon how many privileged instructions are encountered when the request occurs). To establish time intervals greater than 51,200 ϕ clock periods is simply a matter of using the timer interrupt service routine to count the number of interrupts, saving the result in one or more of the scratchpad registers until the desired interval is achieved. With this technique, virtually any time interval, or several time intervals, may be generated.

The timer may be read at any time and in any mode using an input instruction (IN 7 or INS 7) and may take place "on-the-fly" without interfering with normal timer operation. Also, the timer may be stopped at any time by clearing bit 3 of the ICP. The timer holds its current contents indefinitely and resumes counting when bit 3 is again set. Recall, however, that the prescaler is reset whenever the timer is stopped; thus, a series of starting and stopping results in a cumulative truncation error.

A summary of other timer errors is given in the timing section. For a free-running timer in the interval timer mode, the time interval between any two interrupt requests may be in error by ± 6 ϕ clock periods, although the cumulative error over many intervals is zero. The prescaler and timer generate precise intervals for setting the timer interrupt request latch, but the time-out may occur at any time within a machine cycle. (There are two types of machine cycles: short cycles, which consist of 4 ϕ clock periods, and long cycles, which consist of 6 ϕ clock periods.) In the multi-chip F8 family, there is a signal called the write clock that corresponds to a machine cycle. Interrupt requests are synchronized with the internal write clock, thus giving rise to the possible ± 6 ϕ error. Additional errors may arise due to the interrupt request occurring while a privileged instruction or multi-cycle instruction is being executed. Nevertheless, for most applications all of the above errors are negligible, especially if the desired time interval is greater than 1 ms.

Pulse Width Measurement Mode—When ICP bit 4 is set (logic 1) and at least one prescale bit is set, the timer operates in the pulse width measurement mode. This mode is used for accurately measuring the duration of a pulse applied to the EXT INT pin. The timer is stopped and the prescaler is reset whenever EXT INT is at its inactive level. The active level of EXT INT is defined by ICP bit 2: if cleared, EXT INT is active LOW; if set, EXT INT is active high. If ICP bit 3 is set, the prescaler and timer start counting when EXT INT transitions to the active level. When EXT INT returns to the inactive level, the timer stops, the prescaler resets, and, if ICP bit 0 is set, an external interrupt request latch is set. (Unlike timer interrupts, external interrupts are not latched if the ICP interrupt enable bit is not set.)

As in the interval timer mode, the timer may be read at any time and may be stopped at any time by clearing ICP bit 3; the prescaler and ICP bit 1 function as previously described, and the timer still functions as an 8-bit binary down counter, with the timer interrupt request latch being set on the timer transition from H '01' to H 'N'. Note that the EXT INT pin has nothing to do with loading the timer; its action is that of automatically starting and stopping the timer and of generating external interrupts. Pulse widths longer than the prescale value times the modulo-N value are easily measured by using the timer interrupt service routine to store the number of timer interrupts in one or more scratchpad registers.

As for accuracy, the actual pulse duration is typically slightly longer than the measured value because the status of the prescaler is not readable and is reset when the timer is stopped. Thus, for maximum accuracy, it is advisable to use a small division setting for the prescaler.

Event Counter Mode—When ICP bit 4 is cleared and all prescale bits (ICP bits 5, 6, and 7) are cleared, the timer operates in the event counter mode. This mode is used for counting pulses applied to the EXT INT pin. If ICP bit 3 is set, the timer decrements on each transition from the inactive level to the active level of the EXT INT pin. The prescaler is not used in this mode, but, as in the other two timer modes, the timer may be read at any time and may be stopped at any time by clearing ICP bit 3; ICP bit 1 functions as previously described, and the timer interrupt request latch is set on the timer transition from H '01' to H 'N'.

Normally, ICP bit 0 should be kept cleared in the event counter mode; otherwise, external interrupts are generated on the transition from the inactive level to the active level of the EXT INT pin.

For the event counter mode, the minimum pulse width required on EXT INT is 2 ϕ clock periods and the minimum inactive time is 2 ϕ clock periods; therefore, the maximum repetition rate is 500 Hz.

External Interrupts
When the timer is in the interval timer mode, the EXT INT pin is available for non-timer-related interrupts. If ICP bit 0 is set, an external interrupt request latch is set when there is a transition from the inactive level to the active level of EXT INT. (EXT INT is an edge-triggered input.) The interrupt request is latched either until acknowleged by the CPU section or until ICP bit 0 is cleared (unlike timer interrupt requests, which remain latched even when ICP bit 1 is cleared). External interrupts are handled in the same fashion when the timer is in the pulse width measurement mode or in the event counter mode, except that only in the pulse width measurement mode is the external interrupt request latch set on the trailing edge of EXT INT, that is, on the transition from the active level to the inactive level.

Interrupt Handling
When either a timer or an external interrupt request is communicated to the CPU section of the F38E70, it is acknowledged and processed at the completion of the first non-privileged instruction if the interrupt control bit of the status register is set. If the interrupt control bit is not set, the interrupt request continues until either the interrupt control bit is set and the CPU section acknowledges the interrupt or the interrupt request is cleared as previously described.

If there are both a timer interrupt request and an external interrupt request when the CPU section starts to process the requests, the timer interrupt is handled first.

When an interrupt is allowed, the CPU section requests that the interrupting element pass its interrupt vector address to the program counter via the data bus. The vector address for a timer interrupt is H '020'. The vector address for external interrupts is H '0A0'. After the vector address is passed to the program counter, the CPU section sends an acknowledge signal to the appropriate interrupt request latch, which clears that latch. The execution of the interrupt service routine then commences. The return address of the original program is automatically saved in the stack register, P.

Power-on Clear
When power is applied to the F38E70, the program counter and the ICB bit of the status register are cleared. Ports 4, 5, 6, and 7 are loaded with H '00' (thus, the I/O pins for ports 4 and 5 are at V_{OH}). The contents of other registers and ports are undefined. The first program instruction is then fetched from EPROM location H '000'.

External Reset
When $\overline{\text{RESET}}$ is taken LOW, the content of the program counter is pushed to the stack register, and the program counter and the ICB bit of the status register are then cleared. The original stack register content is lost. As with power-on clear, ports 4, 5, 6, and 7 are loaded with H '00'. The contents of all other registers and ports are unchanged. When $\overline{\text{RESET}}$ is taken HIGH, the first program instruction is fetched from EPROM location H '000'.

Test Logic
Special test logic is implemented to allow access to the internal main data bus for test purposes.

In normal operation, the TEST pin is unconnected or is connected to GND. When TEST is placed at a TTL level (2.0 V to 2.6 V), port 4 becomes an output of the internal data bus and port 5 becomes a wired-OR input to the internal data bus. The data appearing on the port 4 pins is logically true, whereas input data forced on port 5 must be logically false. When TEST is placed at a HIGH level (6.0 V to 7.0 V), the ports act as above and, additionally, the 2K x 8 program ROM is prevented from driving the data bus. In this mode, operands and

instructions may be forced externally through port 5 instead of being accessed from the program ROM. When TEST is in either the TTL state or the HIGH state, $\overline{\text{STROBE}}$ ceases its normal function and becomes a cycle clock (identical to the F8 multi-chip system write clock, except inverted).

Timing complexities render the capabilities associated with the TEST pin impractical for use in a user's application, but these capabilities are sufficient to enable Fairchild to implement a rapid method for thoroughly testing the F38E70.

F38E70 Clocks

The time base for the F38E70 may originate in one of four external sources. The four external configurations are shown in *Figure 5*. There is an internal 20 pF capacitor between XTL_1 and GND, and also between XTL_2 and GND. Thus, external capacitors are not required. In all external clock modes, the external time base frequency is divided by two to form the internal ϕ clock.

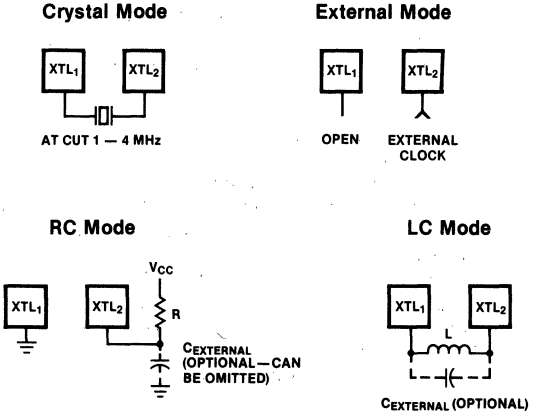
Instruction Set

The F38E70 executes the entire instruction set of the multi-chip F8 family (F3850 family), as shown in *Table 2*. Of course, the STORE instruction is of little use in the F38E70 because only read-only memory exists in the addressing range of the data counter (the data counter, however, is incremented if STORE is executed).

A summary of programmable registers and ports is given in *Figure 6*, followed by a summary of the F38E70 (F8-compatible) instruction set.

Also, for convenient reference, a programming model of the F38E70 is given in *Figure 7*.

Fig. 5 Clock Configurations



Minimum $R = 4k\Omega$

$C = 20.5\ pF \pm 2.5\ pF + C_{EXTERNAL}$

$f_{MIN} \cong \frac{1}{1.1\ RC + 65\ ns}$

$f_{MAX} \cong \frac{1}{1.0\ RC + 15\ ns}$

Example with $C_{EXTERNAL} = 0$

$R = 15\ k\Omega \pm 5\%$

$f \cong 2.9\ MHz \pm 26\%$

Minimum $L = 0.1$ mH

Minimum $Q = 40$

Maximum $C_{EXTERNAL} = 30$ pF

$C = 10\ pF \pm 1.3\ pF + C_{EXTERNAL}$

$f \cong \frac{1}{2\pi\sqrt{LC}}$

Example with $C_{EXTERNAL} = 0$

$L = 0.3\ mH \pm 10\%$

$f \cong 3.0\ MHz \pm 10\%$

Table 2 F38E70 Instruction Set

Accumulator Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Carry	LNK		ACC←(ACC)+CRY	19	1	1	1/0	1/0	1/0	1/0
Add Immediate	AI	ii	ACC←(ACC) Λ H'ii'	24 ii	2	2.5	1/0	1/0	1/0	1/0
AND Immediate	NI	ii	ACC←(ACC) H'ii'	21 ii	2	2.5	0	1/0	0	1/0
Clear	CLR		ACC←H'00'	70	1	1				
Compare Immediate	CI	ii	H'ii'+($\overline{ACC}$)+1	25 ii	2	2.5	1/0	1/0	1/0	1/0
Complement	COM		ACC←(ACC)⊕H'FF'	18	1	1	0	1/0	0	1/0
Exclusive OR Immediate	XI	ii	ACC←(ACC)⊕H'ii'	23 ii	2	2.5	0	1/0	0	1/0
Increment	INC		ACC←(ACC)+1	1F	1	1	1/0	1/0	1/0	1/0
Load Immediate	LI	ii	ACC←H'ii'	20 ii	2	2.5	—	—	—	—
Load Immediate Short	LIS	i	ACC←H'0i'	7 i	1	1	—	—	—	—
OR Immediate	OI	ii	ACC←(ACC) V H'ii'	22 ii	2	2.5	0	1/0	0	1/0
Shift Left One	SL	1	SHIFT LEFT 1	13	1	1	0	1/0	0	1/0
Shift Left Four	SL	4	SHIFT LEFT 4	15	1	1	0	1/0	0	1/0
Shift Right One	SR	1	SHIFT RIGHT 1	12	1	1	0	1/0	0	1
Shift Right Four	SR	4	SHIFT RIGHT 4	14	1	1	0	1/0	0	1

Branch Instructions

(In All Conditional Branches, P0 (P0)+2 if the Test Conditions Are Not Met. Execution Is Complete in 30 Cycles.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Branch on Carry	BC	aa	P0←[(P0)+1]+H'aa' if CRY=1	82 aa	2	3/3.5**	—	—	—	—
Branch on Positive	BP	aa	P0←[(P0)+1]+H'aa' if SIGN=1	81 aa	2	3/3.5**	—	—	—	—
Branch on Zero	BZ	aa	P0←[(P0)+1]+H'aa' if ZERO=1	84 aa	2	3/3.5**	—	—	—	—
Branch on True	BT	t,aa	P0←[(P0)+1]+H'aa' if any test is true	8t aa	2	3/3.5**	—	—	—	—
			t=TEST CONDITION							
			2^2 | 2^1 | 2^0							
			ZERO | CRY | SIGN							
Branch if Negative	BM	aa	P0←[(P0)+1]+H'aa' if SIGN=0	91 aa	2	3/3.5**	—	—	—	—
Branch if No Carry	BNC	aa	P0←[(P0)+1]+H'aa' if CARRY≠0	92 aa	2	3/3.5**	—	—	—	—
Branch if No Overflow	BNO	aa	P0←[(P0)+1]+H'aa' if OVF=0	98 aa	2	3/3.5**	—	—	—	—
Branch if Not Zero	BNZ	aa	P0←[(P0)+1]+H'aa' if ZERO=0	94 aa	2	3/3.5**	—	—	—	—
Branch if False Test	BF	t,aa	P0←[(P0)+1]+H'aa' if all false test bits	9t aa	2	3/3.5**	—	—	—	—
			t=TEST CONDITION							
			2^3 | 2^2 | 2^1 | 2^0							
			OVF | ZERO | CRY | SIGN							
Branch if ISAR (Lower) 7	BR7	aa	P0←[(P0)+1]+H'aa' if ISARL≠7	8F aa	2	2.5	—	—	—	—
			P0←(P0)+2 if ISARL=7			2.0	—	—	—	—
Branch Relative	BR	aa	P0←[(P0)+1]+H'aa'	90 aa	2	3.5	—	—	—	—
Jump*	JMP	aaaa	P0←H'aaaa'	29 aaaa	3	5.5	—	—	—	—

Memory Reference Instructions (In All Memory Reference Instructions, the Data Counter Is Incremented DC←DC•1.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Binary	AM		ACC←(ACC)+[(DC)]	88	1	2.5	1/0	1/0	1/0	1/0
Add Decimal	AMD		ACC←(ACC)+[(DC)]	89	1	2.5	1/0	1/0	1/0	1/0
AND	NM		ACC←(ACC) Λ [(DC)]	8A	1	2.5	0	1/0	0	1/0
Compare	CM		[(DC)]+($\overline{ACC}$)+1	8D	1	2.5	1/0	1/0	1/0	1/0
Exclusive OR	XM		ACC←(ACC)⊕[(DC)]	8C	1	2.5	0	1/0	0	1/0
Load	LM		ACC←[(DC)]	16	1	2.5	—	—	—	—
Logical OR	OM		ACC←(ACC) V [(DC)]	8B	1	2.5	0	1/0	0	1/0
Store	ST		(DC)←(ACC)	17	1	2.5	—	—	—	—

*Privileged instruction

** 3.5 cycles if branch taken.

Note

JMP and PI change accumulator contents to the high byte address.

Table 2 F38E70 Instruction Set (Cont.)

Address Register Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add to Data Counter	ADC		DC←(DC)+(ACC)	8E	1	2.5	—	—	—	—
Call to Subroutine	PK*		P←(P0); P0U←(r12); PL←(r13)	0C	1	4	—	—	—	—
Call to Subroutine Immediate	PI*	aaaa	P←(P); P0←H'aaaa'‡	28 aaaa	3	6.5	—	—	—	—
Exchange DC	XDC		DC—DC1	2C	1	2	—	—	—	—
Load Data Counter	LR	DC,Q	DCU←(r14); DCL←(r15)	0F	1	4	—	—	—	—
Load Data Counter	LR	DC,H	DCU←(r10); DCL←(r11)	10	1	4	—	—	—	—
Load DC Immediate	DCI	aaaa	DC←H'aaaa'	2A aaaa	3	6	—	—	—	—
Load Program Counter	LR	P0,Q	P0U←(r14); P0L←(r15)	0D	1	4	—	—	—	—
Load Stack Register	LR	P,K	PU←(r12); PL←(r13)	09	1	4	—	—	—	—
Return From Subroutine	POP*		P0←(P)	1C	1	2	—	—	—	—
Store Data Counter	LR	Q,DC	r14←(DCU); r15←(DCL)	0E	1	4	—	—	—	—
Store Data Counter	LR	H,DC	r10←(DCU); r11←(DCL)	11	1	4	—	—	—	—
Store Stack Register	LR	K,P	r12←(PU); r13←(P)	08	1	4	—	—	—	—

Scratchpad Register Instructions (Refer to Scratchpad Addressing Modes.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Binary	AS	r	ACC←(ACC)+(r)	Cr	1	1	1/0	1/0	1/0	1/0
Add Decimal	ASD	r	ACC←(ACC)+(r)	Dr	1	2	1/0	1/0	1/0	1/0
Decrement	DS	r	r←(r)+H'FF'	3r	1	1.5	1/0	1/0	1/0	1/0
Load	LR	A,r	ACC←(r)	4r	1	1	—	—	—	—
Load	LR	A,KU	ACC←(r12)	00	1	1	—	—	—	—
Load	LR	A,KL	ACC←(r13)	01	1	1	—	—	—	—
Load	LR	A,QU	ACC←(r14)	02	1	1	—	—	—	—
Load	LR	A,QL	ACC←(r15)	03	1	1	—	—	—	—
Load	LR	r,A	r←(ACC)	5r	1	1	—	—	—	—
Load	LR	KU,A	r12←(ACC)	04	1	1	—	—	—	—
Load	LR	KL,A	r13←(ACC)	05	1	1	—	—	—	—
Load	LR	QU,A	r14←(ACC)	06	1	1	—	—	—	—
Load	LR	QL,A	r15←(ACC)	07	1	1	—	—	—	—
AND	NS	r	ACC←(ACC) Δ (r)	Fr	1	1	0	1/0	0	1/0
Exclusive OR	XS	r	ACC←(ACC)⊕(r)	Er	1	1	0	1/0	0	1/0

Miscellaneous Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Disable Interrupt	DI		RESET ICB	1A	1	2	—	—	—	—
Enable Interrupt*	EI		SET ICB	1B	1	2	—	—	—	—
Input	IN	aa	ACC←(INPUT PORT aa)	26 aa	2	4	0	1/0	0	1/0
Input Short	INS	a	ACC←(INPUT PORT a)	Aa	1	4***	0	1/0	0	1/0
Load ISAR	LR	IS,A	ISAR←(ACC)	0B	1	1	—	—	—	—
Load ISAR Lower	LISL	a	ISARL←a	01101a**	1	1	—	—	—	—
Load ISAR Upper	LISU	a	ISARU←a	01100a**	1	1	—	—	—	—
Load Status Register*	LR	W,J	W←(r9)	1D	1	2	1/0	1/0	1/0	1/0
No-Operation	NOP		P0←(P0)+1	2B	1	1	—	—	—	—
Output	OUT	aa	OUTPUT PORT aa←(ACC)	27 aa	2	4	—	—	—	—
Output Short	OUTS	a	OUTPUT PORT a←(ACC)	Ba	1	4***	—	—	—	—
Store ISAR	LR	A,IS	ACC←(ISAR)	0A	1	1	—	—	—	—
Store Status Register	LR	J,W	r9←(W)	1E	1	1	—	—	—	—

*Privileged instruction
**3-bit octal digit
***Two machine cycles for CPU ports
‡Contents of ACC destroyed

Notes
Each lower case character represents a hexadecimal digit.
Each cycle equals four machine clock periods.
Lower case denotes variables specified by the programmer.

Function Definitons

←	is replaced by
( )	the contents of
(—)	binary ones complement of
+	arithmetic add (binary or decimal)
⊕	logical OR exclusive
Λ	logical AND
V	logical OR inclusive
H'#'	hexadecimal digit

Register Names

a	address variable
A	accumulator
DC	data counter (indirect address register)
DC1	data counter #1 (auxiliary data counter)
DCL	least significant eight bits of data counter addressed
DCU	most significant eight bits of data counter addressed
H	scratchpad register #10 and #11
i and ii	immediate operand
ICB	interrupt control bit
IS	indirect scratchpad address register
ISAR	indirect scratchpad address register
ISARL	least significant three bits of ISAR
ISARU	most significant three bits of ISAR
J	scratchpad register #9
K	registers #12 and #13
KL	register #13
KU	register #12
P0	program counter
P0L	least significant eight bits of program counter
P0U	most significant eight bits of program counter
P	stack register
PL	least significant eight bits of program counter
PU	most significant eight bits of active stack register
Q	registers #14 and #15
QL	register #15
QU	register #14
r	scratchpad register (any address through 11)
W	status register

Scratchpad Addressing Modes (Machine Code Format)

r = C	(hexadecimal) register addressed by ISAR (unmodified)
r = D	(hexadecimal) register addressed by ISAR; ISARL incremented
r = E	(hexadecimal) register addressed by ISAR; ISARL decremented
r = F	(no operation performed)
r = 0-B	(hexadecimal) register 0 through 11 addressed directly from the instruction

Status Register

—	no change in condition
1/0	is set to 1 or 0, depending on conditions
CRY	carry flag

4

EPROM Programming

When V_{PP} is applied to the TEST 1 pin, the device goes into the program or verify mode and the I/O ports take on the different functions of $\overline{\text{DATA IN}}$ (port 5), DATA OUT (port 4), EPROM address (11 pins of ports 0 and 1), and $\overline{\text{PROG}}$ (port 1_6). The verify mode exists when $\overline{\text{PROG}}$ is HIGH and TEST 2 = V_{CC}. Port 4 outputs the data content of the EPROM according to the address A_0 through A_{10}. The logical sense is true, and for an unprogrammed location, the outputs are high. During verify mode, the data on port 5 has no effect on the port 4 output. The program mode exists when $\overline{\text{PROG}}$ is taken low. All addresses and DATA IN must be stable before going into this mode. During this mode, the data appearing on port 5 is "burned in" to the EPROM. Note that the sense of port 5 is logically false. At the same time, port 4 outputs the data on the internal data bus, which is exactly equal to the inversion of the data going in on port 5. Port 4 does not indicate satisfactory completion of the EPROM programming in the program mode. The $\overline{\text{PROG}}$ pin must be high before the V_{PP} is applied in order to prevent a programming error.

CAUTION
Applying V_{PP} to the TEST 1/V_{PP} pin without the presence of V_{CC} will damage the device.

F38E70 Erasing Instructions

The contents of the F38E70 EPROM can be erased by exposure to high-intensity shortwave ultraviolet (UV) light with a wavelength of 2537 Angstroms (Å). This can be accomplished with ultraviolet light EPROM erasure devices that are available from several U.S. manufacturers. These erasure devices contain a UV light source, which is usually placed approximately 1 or 2 inches from the EPROM so that the transparent window on top of the device is illuminated. The minimum required integrated dose (intensity x exposure time) of UV light energy incident on the window of the device in order to reliably ensure complete erasure is 15 watt-sec/cm^2. The UV erasure unit should be periodically calibrated if minimum exposure times are to be used. (Minimum exposure times range from 10 to 45 minutes, depending on the model type and age of UV lamp.) If longer exposure times are possible, variations in the output light intensity of the UV light source are not critical.

Fig. 6 Programmable Registers and Ports

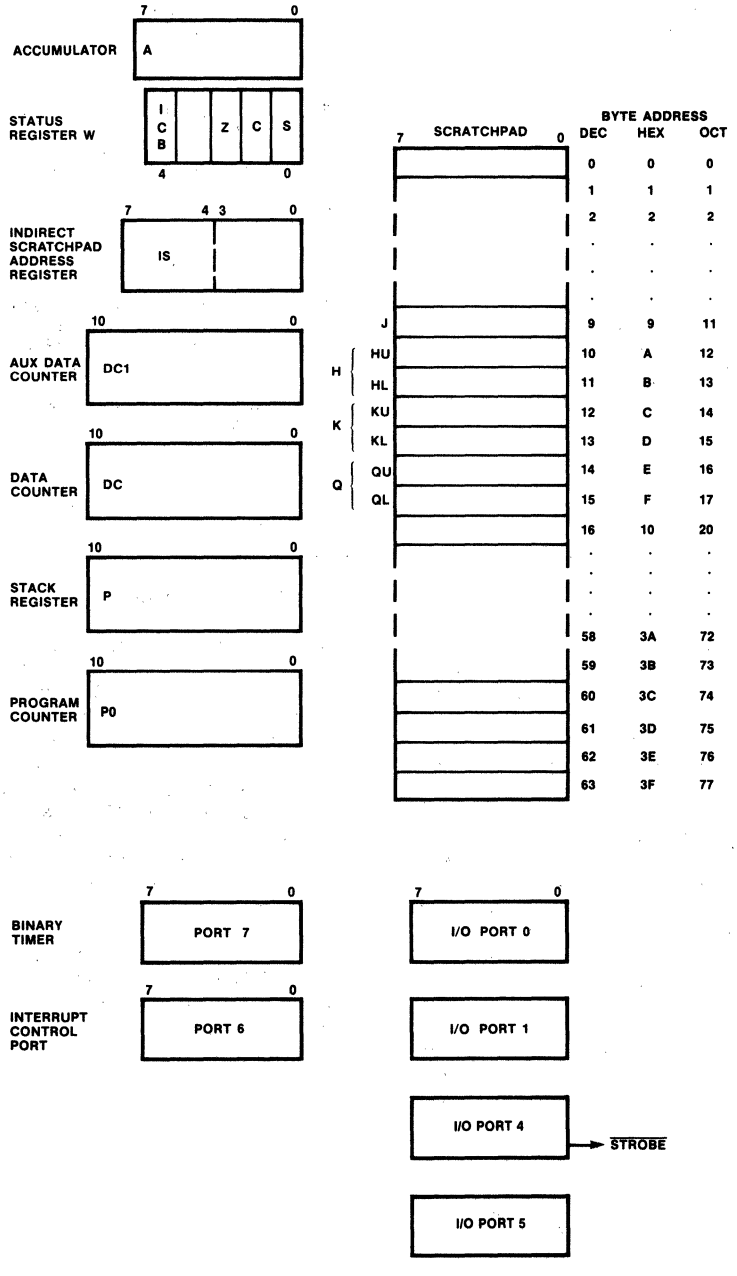
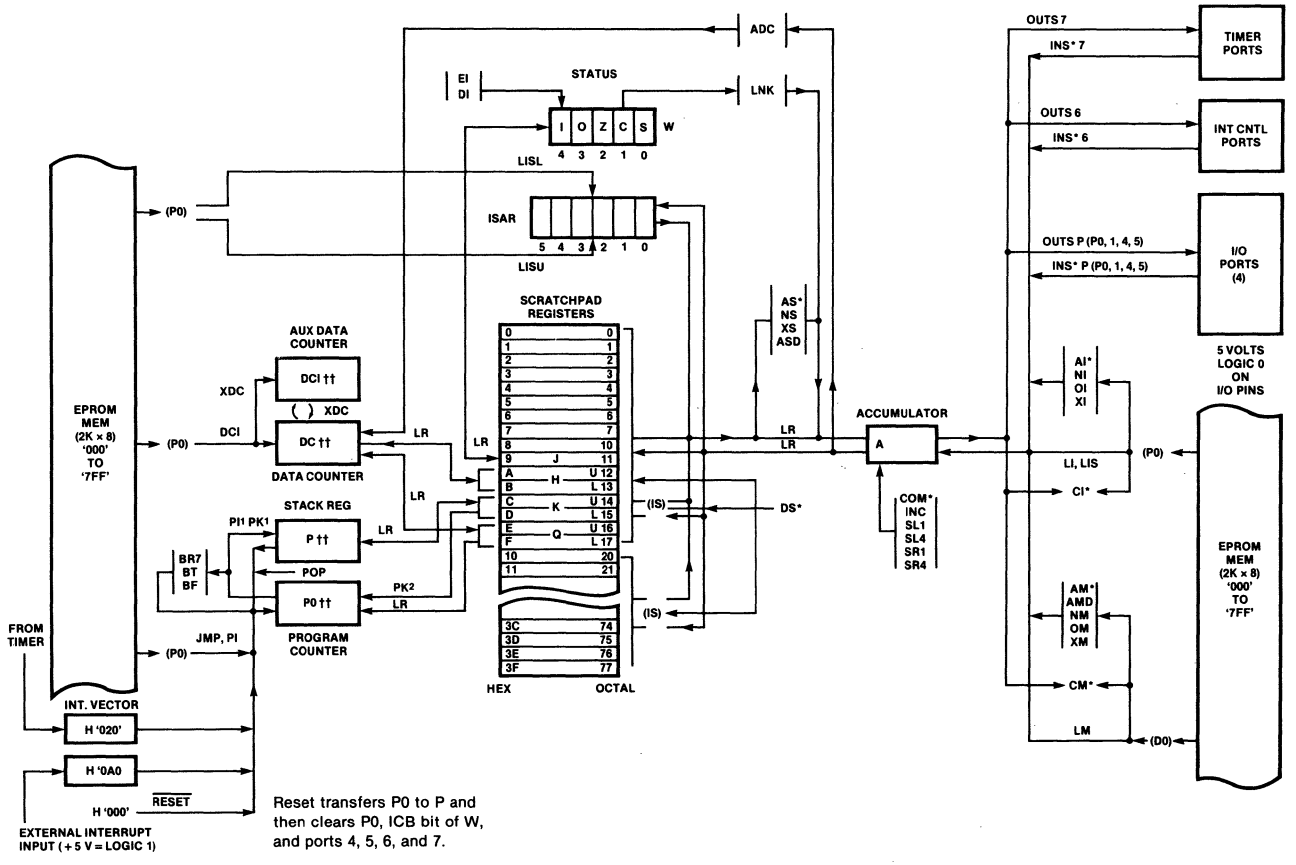


Fig. 7 Programming Model



Reset transfers P0 to P and then clears P0, ICB bit of W, and ports 4, 5, 6, and 7.

Note

The instructions PI and PK are shown in two sequential parts (PI_1, PI_2, and PK_1, PK_2).

*These instructions set status.

†The value of the external interrupt input is loaded to bit 7 of the accumulator (with bits 0 through 6 loaded with zeros) when the instruction 'INS 6' is executed. This instruction also sets status.

††P0, P, DC, and DC1 are 11 bit registers.

Supplementary Notes

For total software compatibility when expanding into a multi-chip configuration, the F3871 Peripheral Input/Output circuit should be used. The F3871 has the same improved timer (binary count, readable, and three modes of operation) and ready strobe output as the F38E70.

The interrupt control bit of the status register is automatically reset when an interrupt request is acknowledged. It is then the programmer's responsibility to determine when ICB is again set (by executing an EI instruction). This action prevents an interrupt service routine from being interrupted unless the programmer so desires.

When reading the interrupt control port (port 6), bit 7 of the accumulator is loaded with the actual logic level being applied to the EXT INT pin, regardless of the status of ICP bit 2 (the EXT INT active level bit); that is, if EXT INT is at +5 V, bit 7 of the accumulator is set to a logic 1, but if EXT INT is at GND, accumulator bit 7 is reset to logic 0.

In the "F38E70 (F8-compatible) Instruction Set" summary, the number of cycles shown is "nominal" machine cycles. A nominal machine cycle is defined as 4 ϕ clock periods, thus requiring 2 μs for a 2 MHz ϕ clock frequency (4 MHz external time base frequency).

Also, the summary uses the following nomenclature for register names:

F8	F38E70	
PC_0 =	P0	Program Counter
PC_1 =	P	Stack Register
DC_0 =	DC	Data Counter
DC_1 =	DC1	Auxiliary Data Counter

This nomenclature is used in order to be consistent with the assembly language mnemonics.

For the F38E70, execution of an INS or OUTS instruction requires two machine cycles for ports 0 and 1, while ports 4 and 5 require four machine cycles.

When an external reset of the F38E70 occurs, P0 is pushed into P and the old contents of P are lost. It must be noted that an external reset is recognized at the start of the machine cycle and not necessarily at the end of an instruction. Thus, if the F38E70 is executing a multi-cycle instruction, that instruction is not completed and the contents of P upon reset may not necessarily be the address of the instruction that would have been executed next. It may, for example, point to an immediate operand if the reset occurred during the second cycle of an LI or CI instruction. Additionally, several instructions (JMP, PI, PK, LR, P0, Q) as well as the interrupt acknowledge sequence modify P0 in parts. That is, they alter P0 by loading first one part, then the other, and the entire operation takes more than one cycle. Should reset occur during this modification process, the value pushed into P is part of the old P0 (the as-yet unmodified part) and part of the new P0 (the already modified part). Thus, care should be taken (perhaps by external gating) to ensure that reset does not occur at an undesirable time if any significance is to be given to the contents of P after a reset occurs.

Absolute Maximum Ratings

The absolute maximum ratings of the F38E70 are as follows:

Temperature (Ambient Under Bias)	0°C, +70°C
Storage Temperature	−55°C, +150°C
Voltage on any Pin with Respect to Ground (Except Test Pin)	−1.0 V, +7 V
Test Pin Voltage with Respect to V_{SS}	−1.0 V, +27 V

CAUTION

Applying V_{PP} to the TEST 1/V_{PP} pin without the presence of V_{CC} will damage the device.

Power Dissipation	1.0 W

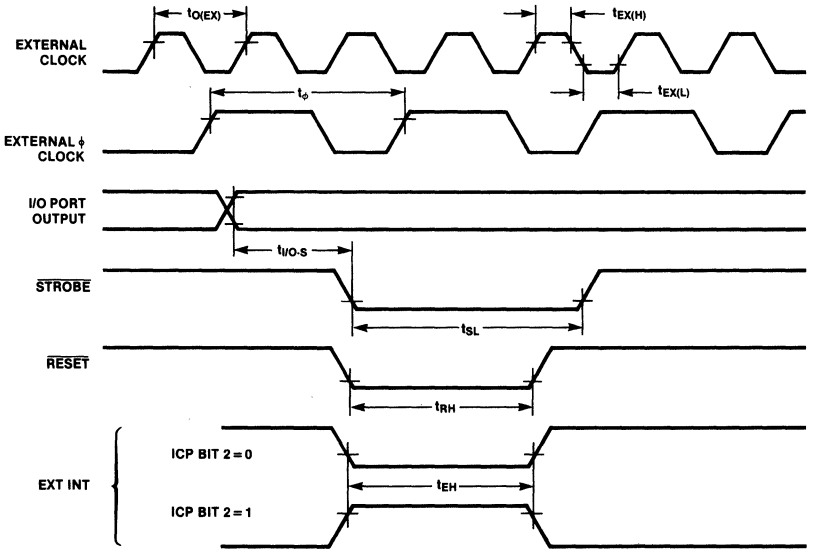
These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Timing Characteristics $V_{CC} = +5\ V \pm 10\%$, $T_A = 0°C$ to $+70°C$

Signal	Symbol	Characteristic	Min	Max	Unit	Comments (Note 3)
XTL_1 XTL_2	t_0(XTL)	Time Base Period, Crystal Mode	250	5000	ns	4 MHz–2 MHz
	t_0(LC)	Time Base Period, LC Mode	250	5000	ns	4 MHz–2 MHz
	t_0(RC)	Time Base Period, RC Mode	250	5000	ns	4 MHz–2 MHz
	t_0(EX)	Time Base Period, External Mode	250	5000	ns	4 MHz–2 MHz
	t_{EX}(H)	External Clock Pulse Width, High	90	t_0(EX)-100	ns	
	t_{EX}(L)	External Clock Pulse Width, Low	90	t_0(EX)-100	ns	
ϕ	$t\phi$	Internal ϕ Clock Period	$2t_0$ typ.		ns	0.5 μs at 4 MHz ext. time base
$\overline{STROBE}$	$t_{I/O-S}$	Port Output to $\overline{STROBE}$ Delay	$3t\phi - 1000$ min $3t\phi + 250$ max		ns	Note 1
	t_{SL}	$\overline{STROBE}$ Pulse Width, Low	$8t\phi - 250$ min $12t + 250$ max		ns	
$\overline{RESET}$	t_{RH}	$\overline{RESET}$ Hold Time, Low	$6t\phi + 750$ min		ns	
EXT INT	t_{EH}	EXT INT Hold Time, Active State	$6t\phi + 750$ min		ns	Note 2

Notes
1. Load is 50 pF plus 1 standard TTL input.
2. Specification is applicable when the timer is in the interval timer mode. See "Timer Characteristics" for EXT INT requirements when in the pulse width measurement mode or the event counter mode.
3. The timing diagrams are given in *Figure 8*.

Fig. 8 Timing Diagrams



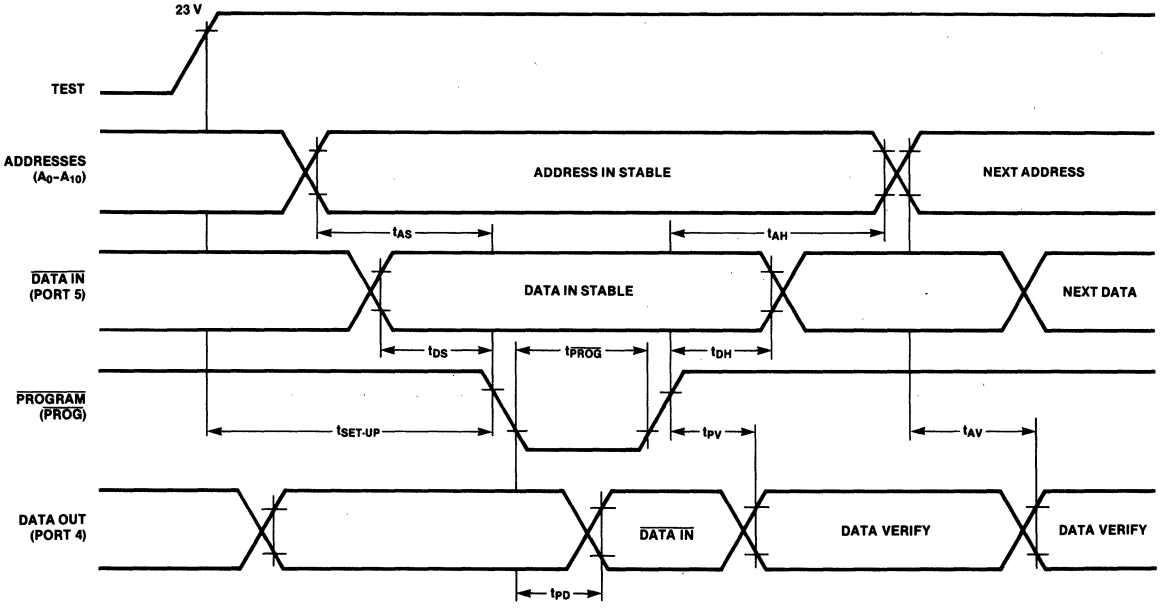
Note
All measurements are referenced to V_{IL} max., V_{IH} min., V_{OL} max., or V_{OH} min.

Program/Verify Timing

Symbol	Parameter	Min	Max	Unit
t_{SET-UP}	23 V Applied to $\overline{PROG}$	5		μs
t_{AS}	Address Set-up Time	1		μs
t_{AH}	Address Hold Time	1		μs
t_{DS}	Data Set-up Time	1		μs
t_{DH}	Data Hold Time	1		μs
t_{AV}	Address to Data Out in Verify		5	μs
t_{PV}	$\overline{PROG}$ to Data Out in Verify		2	μs
t_{PD}	$\overline{PROG}$ to Data Out in Programming		5	μs
t_{PROG}	Programming Time	50	60	ms

Note
Timing diagrams are given in *Figure 9*.

Fig. 9 Program/Verify Timing Diagrams



DC Characteristics $V_{CC} = 5\ V \pm 10\%$, $T_A = 0°C$ to $+70°C$

Symbol	Characteristic	Min	Typ	Max	Unit	Test Conditions
I_{CC}	Power Supply Current		70	100	mA	Outputs Open
P_D	Power Dissipation		375	550	mW	Outputs Open
V_{IHEX}	External Clock Input High Voltage	2.4		5.8	V	
V_{ILHEX}	External Clock Input Low Voltage	−0.3		0.6	V	
I_{HEX}	External Clock Input High Current			100	μA	$V_{IHEX} = 2.4\ V$
I_{ILEX}	External Clock Input Low Current			−100	μA	$V_{ILEX} = 0.6\ V$
V_{IH}	Input High Voltage	2.0		5.8	V	
V_{IL}	Input Low Voltage	−0.3		0.8	V	
I_{IH}	Input High Current (Except Open-Drain and Direct-Drive I/O Ports)			100	μA	$V_{IH} = 2.4\ V$, Internal Pull-Up
I_{IL}	Input Low Current (Except Open-Drain and Direct-Drive Ports)			−1.6	mA	$V_{IL} = 0.4\ V$
I_{LOD}	Leakage Current (Open-Drain Ports)			10	μA	Pull-Down, Device Off
I_{OH}	Output High Current (Except Open-Drain and Direct-Drive Ports)	−100			μA	$V_{OH} = 2.4\ V$
I_{OHDD}	Output Drive Current (Direct-Drive Ports)	−1.5		−8	mA	$V_{OH} = 0.7\ V$ to $1.5\ V$
I_{OL}	Output Low Current	1.8			mA	$V_{OL} = 0.4\ V$
I_{OHS}	Output High Current ($\overline{STROBE}$ Output)	−300			μA	$V_{OH} = 2.4\ V$
I_{OLS}	Output Low Current ($\overline{STROBE}$ Output)	5.0			mA	$V_{OL} = 0.4\ V$
V_{TEST}	Test Pin Voltage for Program/Verify Mode	23	23.5	24	V	
I_{TEST}	Test Pin Current for Program/Verify Mode		20	30	mA	$V_{\overline{PROG}} = 0.4\ V$, $V_{TEST} = 24\ V$

Capacitance $T_A = 25°C$, $f = 2$ MHz

Symbol	Characteristic	Min	Max	Unit	Test Condition
C_{IN}	Input Capacitance: I/O Ports, $\overline{RESET}$, EXT INT		7	pF	Unmeasured pins returned to GND
C_{XTL}	Input Capacitance: XTL_1, XTL_2	1 8	23	pf	

Typical Thermal Resistance Values

Plastic	
θ_{JA} (Junction to ambient)	60°C/W (still air)
θ_{JA} (Junction to case)	42°C/W
Ceramic	
θ_{JA} (Junction to ambient)	48°C/W (still air)
θ_{JA} (Junction to case)	33°C/W

Ordering Information

Part Number	Package	Temperature Range*
F38E70DC	Ceramic	C
F38E70DL	Ceramic	L
F38E70DM	Ceramic	M
F38E70PC	Plastic	C
F38E70PL	Plastic	L
F38E70PM	Plastic	M

*C = Commercial Temperature Range 0 °C to + 70 °C
L = Limited Temperature Range − 40 °C to + 85 °C
M = Military Temperature Range − 55 °C to + 125 °C

FAIRCHILD
A Schlumberger Company

F38E70-21
Single-Chip Microcomputer

Microprocessor Product

Description

The Fairchild F38E70-21 is virtually identical to the F38E70 single-chip microcomputer. Specification differences between the F38E70 and the F38E70-21 are described in this data sheet (refer to the F38E70 data sheet for a complete device description).

Page 1

F38E70 Features

- **Crystal, LC, RC, External, or Internal Time Base**

F38E70-21 Features

- **Crystal, LC, RC, or External Time Base**

4

Page 3

F38E70 Pin Functions

Pin Name	Type	Description
XTL_1, XTL_2	Input	The time base inputs to which a crystal (1 to 4 MHz), LC network, RC network, or an external single-phase clock may be connected. If timing is not critical, the F38E70 operates from its internal oscillator with no external components.
TEST $1/V_{PP}$	Input	An input used only in testing and programming the F38E70. For normal circuit functionality, this pin is left unconnected or may be grounded. For EPROM programming, the test pin is connected to the programming voltage (typically 23 V).

F38E70-21 Pin Functions

Pin Names	Type	Description
XTL_1, XTL_2	Input	The time base inputs to which a crystal (2 to 4 MHz), LC network, RC network, or an external single-phase clock may be connected.
TEST $1/V_{PP}$	Input	An input used only in testing and programming the F38E70- 21. For normal circuit functionality, this pin is left unconnected or may be grounded. For EPROM programming, the test pin is connected to the programming voltage (typically 23.5 V).

Pages 9 and 10

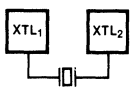
F38E70-21 Test Logic

All references to "TEST pin" should be changed to "TEST 1 pin".

Page 10

F38E70 Figure 5 Clock Configurations

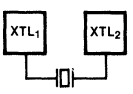
Crystal Mode



AT CUT 1 — 4 MHz

F38E70-21 Figure 5 Clock Configurations

Crystal Mode



AT CUT 2 — 4 MHz

Page 13

F38E70 EPROM Programming

When 23 V is applied to the TEST 1 pin,

CAUTION
Applying + 25 V to the V_{PP} pin without

F38E70-21 EPROM Programming

When 23 V to 24 V is applied to the TEST 1 pin,

CAUTION
Applying + 23 V to the V_{PP} pin without

Page 16

F38E70 Absolute Maximum Ratings

Test Pin Voltage with Respect to V_{SS} − 1.0 V, + 27 V

CAUTION
Applying + 25 V to the V_{PP} pin without

F38E70-21 Absolute Maximum Ratings

Test Pin Voltage with Respect to V_{SS} 1.0 V, + 25 V

CAUTION
Applying + 23 V to the V_{PP} pin without

Page 17

F38E70 Timing Characteristics

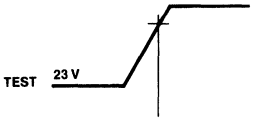
Signal	Symbol	Characteristic	Min	Max	Unit	Comments (Note 3)
XTL_1 XTL_2	t_0(XTL)	Time Base Period, Crystal Mode	250	1000	ns	4 MHz–1 MHz
	t_0(LC)	Time Base Period, LC Mode	250	1000	ns	4 MHz–1 MHz
	t_0(RC)	Time Base Period, RC Mode	250	2000	ns	4 MHz–1 MHz
	t_0(EX)	Time Base Period, External Mode	250	2500	ns	4 MHz–1 MHz
	t_{EX}(H)	External Clock Pulse Width, High	90	2000	ns	
	t_{EX}(L)	External Clock Pulse Width, Low	90	2000	ns	

F38E70-21 Timing Characteristics

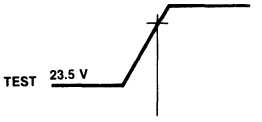
Signal	Symbol	Characteristic	Min	Max	Unit	Comments (Note 3)
XTL_1 XTL_2	t_0(XTL)	Time Base Period, Crystal Mode	250	500	ns	4 MHz–2 MHz
	t_0(LC)	Time Base Period, LC Mode	250	500	ns	4 MHz–2 MHz
	t_0(RC)	Time Base Period, RC Mode	250	500	ns	4 MHz–2 MHz
	t_0(EX)	Time Base Period, External Mode	250	500	ns	4 MHz–2 MHz
	t_{EX}(H)	External Clock Pulse Width, High	90	t_0(EX) − 100	ns	
	t_{EX}(L)	External Clock Pulse Width, Low	90	t_0(EX) − 100	ns	

Page 17

F38E70 Figure 8 Timing Diagrams



F38E70-21 Figure 8 Timing Diagrams



Page 18

F38E70 Program/Verify Timing

Symbol	Parameter	Min	Max	Unit
t_{PROG}	Programming Time	50		ms

F38E70-21 Program/Verify Timing

Symbol	Parameter	Min	Max	Unit
t_{PROG}	Programming Time	50	60	ms

Page 19

F38E70 DC Characteristics

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
I_{CC}	Power Supply Current			60	mA	Outputs Open
P_D	Power Dissipation			330	mW	Outputs Open
⋮	⋮	⋮	⋮	⋮	⋮	⋮
V_{TEST}	Test Pin Voltage for Program/Verify Mode			23 (±.5)	V	
I_{TEST}	Test Pin Current for Program/Verify Mode			20	mA	$V_{\overline{PROG}} = 0.4$ V, $V_{TEST} = 23$ V

F38E70-21 DC Characteristics

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
I_{CC}	Power Supply Current		70	100	mA	Outputs Open
P_D	Power Dissipation		375	550	mW	Outputs Open
⋮	⋮	⋮	⋮	⋮	⋮	⋮
V_{TEST}	Test Pin Voltage for Program/Verify Mode	23	23.5	24	V	
I_{TEST}	Test Pin Current for Program/Verify Mode		20	30	mA	$V_{\overline{PROG}} = 0.4$ V, $V_{TEST} = 24$ V

FAIRCHILD
A Schlumberger Company

F3872/F38L72
Single-Chip Microcomputer

Microprocessor Product

Description

The Fairchild F3872 is a complete 8-bit microcomputer on a single MOS integrated circuit. It can execute the F8 instruction set of more than 70 commands, allowing expansion into multi-chip configurations with software compatibility. The device features 64 bytes of scratchpad RAM, 64 bytes of power-down executable RAM, a programmable binary timer, 32 bits of I/O, a single +5 V power supply requirement, and a choice of 1K, 2K, 3K, or 4K bytes of ROM. A low-power standby option for the executable RAM is available on the F38L72.

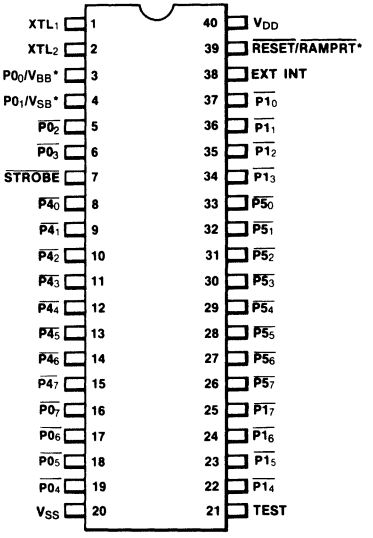
Utilizing Fairchild's double-ion-implanted, N-channel silicon-gate technology and advanced circuit design techniques, the single-chip F3870 offers maximum cost effectiveness in a wide range of control and logic replacement applications.

The F3872 is an expanded memory version of the F3870 single-chip microcomputer. It is identical to the F3870 in the following areas: instruction set, architecture, ac and dc characteristics, and pinout. The only difference between the F3872 and the F3870 lies in the memory expansion and the appropriate memory address registers.

- **Single-chip Microcomputer**
- **Same Pinout as F3870**
- **Software-Compatible with F8 Family**
- **1024-, 2048-, 3072-, or 4032-Byte Mask-Programmable ROM**
- **64-Byte Scratchpad RAM**
- **32-Bit (4-Port) TTL-Compatible I/O**
- **Programmable Binary Timer:**
  - **Interval Timer Mode**
  - **Pulse Width Measurement Mode**
  - **Event Counter Mode**
- **External Interrupt**
- **Crystal, LC, RC, or External Time Base**
- **Low Power (285 mW, Typical)**
- **Single +5 V ± 10% Power Supply**
- **64 Additional Bytes of Executable RAM Addressable by Program or Data Counter**
- **Standby Option for Executable RAM**
  - **Low Standby Power (8.2 mW)**
  - **3.2 V Minimum Standby Supply Voltage**
  - **No External Components Required to Trickle Charge Battery**

Connection Diagram

40-Pin DIP



(Top View)

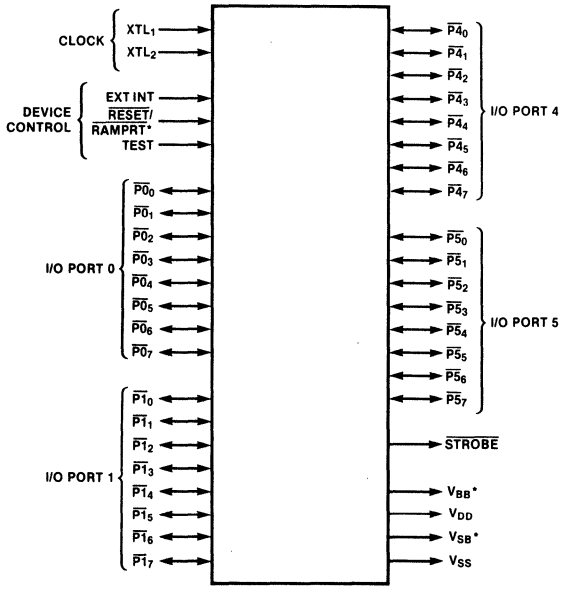
*Programmable pin; function determined by device option (standard or standby mode).

Signal Functions

The functions of the F3872 inputs and outputs are described in *Table 1.*

4

Signal Functions



Device Organization

This section describes the basic functional elements of the F3872 shown in *Figures 1* and *2*.

Main Control Logic

The instruction register (IR) receives the operation code (op code) of the instruction to be executed from the program ROM via the data bus. During all op code fetches, eight bits are latched into the IR. Some instructions are completely specified by the upper four bits of the op code; in such instructions, the lower four bits are an immediate register address or an immediate 4-bit operand. Once latched into the IR, the main control logic decodes the instruction and provides the necessary control gating signals to all circuit elements.

ROM Address Registers

There are four 12-bit registers associated with the program ROM of the F3872. *(In the F3872-1, -2, and -3, the 12-bit registers can address more memory space than is physically available on the chip; user caution is advised.)* These are the program counter (P0), the stack register (P), the data counter (DC), and the auxiliary data counter (DC1). The program counter is used to address instructions or immediate operands. The stack register is used to save the contents of P0 during an interrupt or subroutine call. Thus, P contains the return address at which processing is to resume upon completion of the subroutine or the interrupt routine.

The data counter is used to address data tables. This register is autoincrementing. Of the two data counters, only DC can access the ROM. However, the XDC instruction allows DC and DC1 to be exchanged.

Associated with the F3872 address registers is a 12-bit adder/incrementer. This logic element is used to increment P0 or DC when required and is also used to add displacements to P0 on relative branches or to add the data bus contents to DC in the add data counter (ADC) instruction.

Program ROM

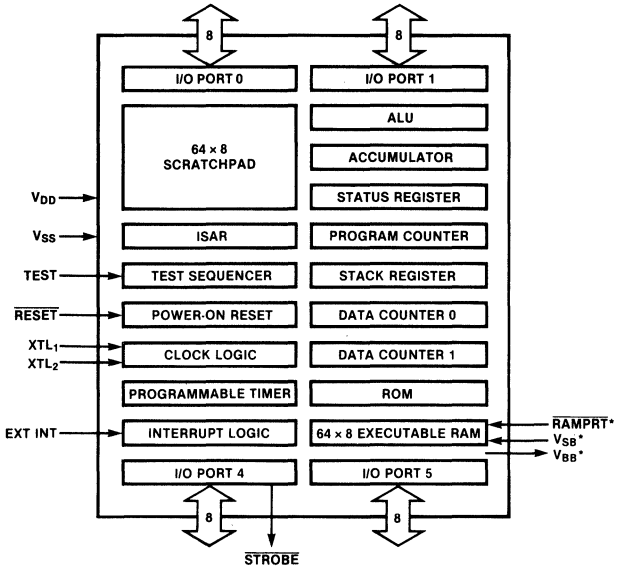
The microcomputer program and data constants are stored in the program ROM, which may be 1024 x 8 (F3872-1), 2048 x 8 (F3872-2), 3072 x 8 (F3872-3), or 4032 x 8 (F3872-4) bytes. When a ROM access is required, the appropriate address register (P0 or DC) is gated onto the ROM address bus and the ROM output is gated onto the main data bus. The first byte in the ROM is location zero.

Table 1 Signal Functions

Mnemonic	Pin No.	Name	Description
Device Control			
EXT INT	38	External Interrupt	Software-programmable input that is also used in conjunction with the timer for pulse width measurement and event counting.
$\overline{\text{RESET}}$/ $\overline{\text{RAMPRT}}$	39	External Reset/RAM Protect	Input that, in standard operating mode, may be used to externally reset the F3872. When pulled low, the F3872 resets; when then allowed to go high, the F3872 begins program execution at program location H'0000'. When RAM standby mode is selected, may be used as RAM protect control. When pulled low, the RAM is disabled and, therefore, protected from any alterations during loss of V_{DD}.
TEST	21	Test Line	An input used only in testing the F3872. For normal circuit operation, TEST is left unconnected or grounded.
Clock			
$\overline{\text{STROBE}}$	7	Ready Strobe	Normally high output that provides a single low pulse after valid data is present on the $\overline{P4_0}$-$\overline{P4_7}$ pins during an output instruction.
XTL_1, XTL_2	1, 2	Time Base	Inputs to which a crystal (1 MHz to 4 MHz), LC network, RC network, or external single-phase clock may be connected.
$\overline{P0_0}$-$\overline{P0_7}$ $\overline{P1_0}$-$\overline{P1_7}$ $\overline{P4_0}$-$\overline{P4_7}$ $\overline{P5_5}$-$\overline{P5_7}$	3-6, 8-19, 22-37	I/O Ports	Thirty-two bidirectional lines that can be individually used as either TTL-compatible inputs or latched outputs; $P0_0$ and $P0_1$ may also serve power outputs in standby mode.
Power			
V_{BB}	3	Substrate Decoupling	Substrate decoupling power pin that is used only when the standby option is selected; a 0.01 μF capacitor is required to provide substrate decoupling; alternative function of $P0_0$, which is the standard function.
V_{DD}	40	Power Input	+5 V ± 10% power supply
V_{SB}	4	Standby Power	The RAM standby power supply if the standby option (+5.5 V to +3.2 V) is selected; alternative function of $P0_1$, which is the standard function.
V_{SS}	20	Ground	Signal and power ground

4

Fig. 1 F3872 Architecture



*Standby Mode Only.

64 x 8 Executable RAM
The upper 64 bytes of the total memory of the F3872 is executable RAM. The first byte is at address 4032 decimal ('FC0' hexadecimal). As with the ROM, the RAM may be accessed by the P0 and DC address registers. It may be written to via the store (ST) instruction, and it may be read from via the load (LM) instruction. Additionally, instructions may be executed from the RAM. A mask-programmable standby power option is available in which the 64 x 8 RAM remains powered and protected so that its contents are saved during a loss of the normal circuit power supply.

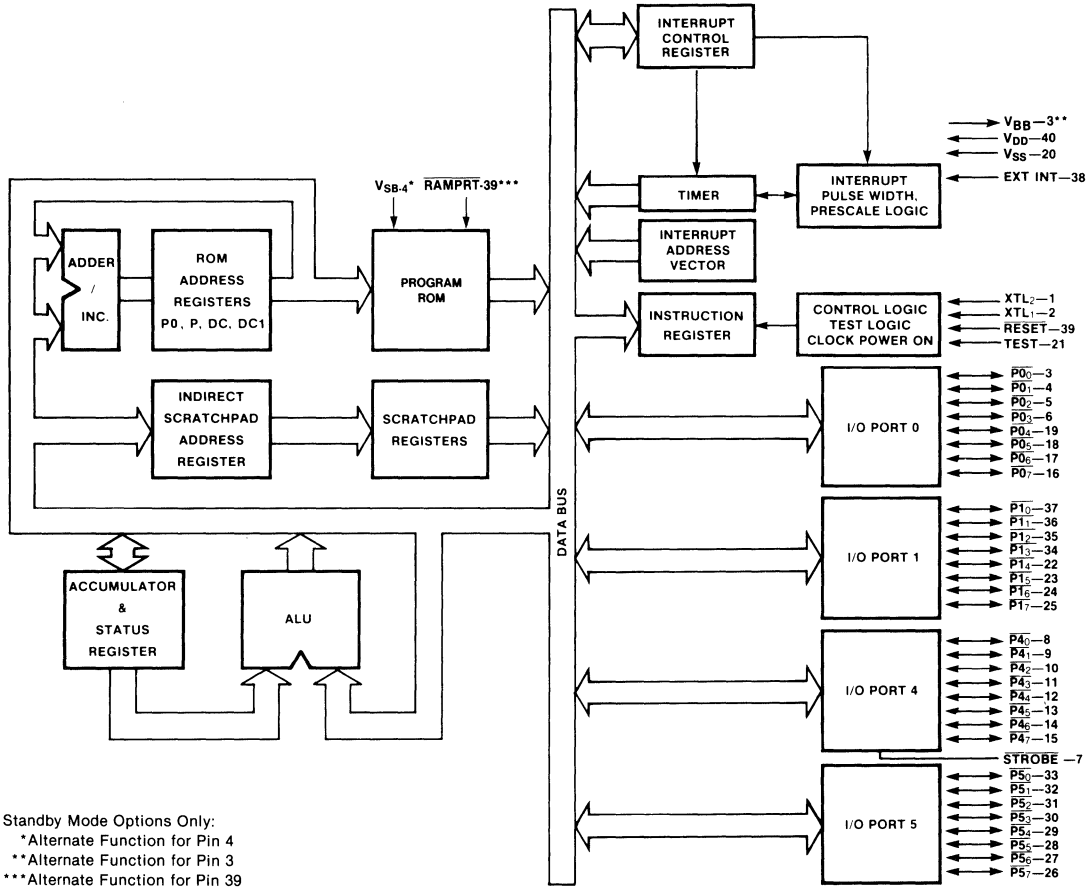
Scratchpad and ISAR
The scratchpad provides 64 8-bit registers that may be used as general-purpose RAM. The indirect scratchpad address register (ISAR) is a 6-bit register used to address the 64 registers. All 64 registers may be accessed using the ISAR. In addition, the lower order 12 registers may also be directly addressed.

The ISAR can be visualized as holding two octal digits. This division of the ISAR is important, since a number of instructions increment or decrement only the least significant three bits of the ISAR when referencing scratchpad bytes via the ISAR. This makes it easy to reference a buffer consisting of contiguous scratchpad bytes. For example, when the low-order octal digit is incremented or decremented, the ISAR is incremented from octal 27 to 20 or is decremented from octal 20 to 27. This feature of the ISAR is very useful in many program sequences. All six bits of the ISAR may be loaded at one time, or either half may be loaded independently.

Scratchpad registers 9 through 15 (decimal) are given mnemonic names (J, H, K, and Q) because of special linkages between these registers and other registers, such as the stack register. These special linkages facilitate the implementation of multi-level interrupts and subroutine nesting. For example, the instruction LR K, P stores the lower eight bits of the stack register in register 13 (K lower, or KL) and stores the upper four bits of P in register 12 (K upper, or KU). The scratchpad is not protected by the standby power option.

Arithmetic and Logic Unit (ALU)
After receiving commands from the main control logic, the ALU performs the required arithmetic or logic operations (using the data presented on the two input

Fig. 2 F3872 Block Diagram



Standby Mode Options Only:
*Alternate Function for Pin 4
**Alternate Function for Pin 3
***Alternate Function for Pin 39

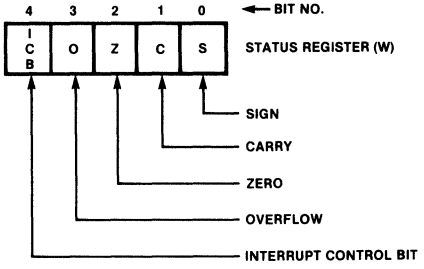
buses) and provides the result on the result bus. The arithmetic operations that can be performed in the ALU are binary add, decimal adjust, add with carry, decrement, and increment. The logic operations that can be performed are AND, OR, exclusive-OR, ones complement, shift right, and shift left. Besides providing the result on the result bus, the ALU also provides four signals presenting the status of the result. These signals, stored in the status register (W), represent the CARRY, OVERFLOW, SIGN, and ZERO conditions of the result of the operation.

Accumulator

The accumulator (ACC) is the principal register for data manipulation within the F3872. The ACC serves as one input to the ALU for arithmetic or logical operation. The results of ALU operations are stored in the ACC.

Status Register

The status register (also referred to as the W register) holds five status flags, as follows:



Summary of Status Bit

$$\text{OVERFLOW} = \text{CARRY}_7 \oplus \text{CARRY}_6$$

$$\text{ZERO} = \overline{ALU_7} \wedge \overline{ALU_6} \wedge \overline{ALU_5} \wedge \overline{ALU_4} \wedge \overline{ALU_3} \wedge \overline{ALU_2} \wedge \overline{ALU_1} \wedge \overline{ALU_0}$$

$$\text{CARRY} = \text{CARRY}_7$$

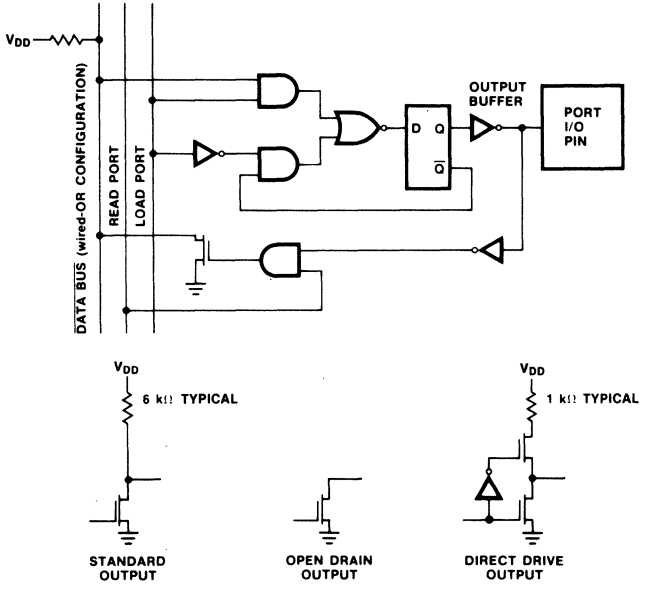
$$\text{SIGN} = \overline{ALU_7}$$

The interrupt control bit (ICB) of the status register may be used to allow or disallow interrupts in the F3872. This bit is not the same as the two interrupt enable bits in the interrupt control port (ICP). If the ICB is set and the F3872 interrupt logic communicates an interrupt request to the CPU section, the interrupt is acknowledged and processed upon completion of the first non-privileged instruction. If the ICB is cleared, an interrupt request is not acknowledged or processed until the ICB is set.

I/O Ports

The F3872 provides four complete bidirectional I/O ports; these are ports 0, 1, 4, and 5. In addition, the interrupt control register is addressed as port 6 and the binary timer is addressed as port 7. An output instruction (OUT or OUTS) causes the contents of the ACC to be latched into the addressed port. An input instruction (IN or INS) transfers the contents of the port to the ACC (port 6 is an exception that is described later). The I/O pins on the F3872 are logically inverted. The schematic of an I/O pin and conceptual illustrations of available output drive options are shown in *Figure 3.*

Fig. 3 I/O Port Diagram



Ports 0 and 1 are standard output type only.
Ports 4 and 5 may be any of the three output options, each pin individually assignable to any port.
The $\overline{\text{STROBE}}$ output is always configured similar to a standard output, except that it is capable of driving three TTL loads.
The $\overline{\text{RESET}}$ and EXT INT pins may have standard 6 kΩ (typical) pull-up or may have no pull-up.

An output ready strobe is associated with port 4. This flag may be used to signal a peripheral device that the F3872 has just completed a single low pulse shortly after the output operation is completely finished, so either edge may be used to signal the peripheral. This $\overline{\text{STROBE}}$ signal may also be used to request new input information from a peripheral simply by doing a dummy output of H'00' to port 4 after completing the input operation.

Timer and Interrupt Control Port

The timer is an 8-bit binary down counter that is software-programmable to operate in one of three modes: the interval timer mode, the pulse width measurement mode, or the event counter mode; the timer characteristics are described in *Table 2.* As shown in *Figure 4,* associated with the timer is an 8-bit register called the interrupt control port, a programmable prescaler, and an 8-bit modulo-N register; a functional logic diagram is shown in *Figure 5.*

The desired timer mode, prescale value, starting and stopping the timer, active level of the EXT INT pin, and local enabling or disabling of interrupts are selected by outputting the proper bit configuration from the accumulator to the ICP (port 6) with an OUT or OUTS instruction. Bits within the ICP are defined as follows:

Interrupt Control Port (Port 6)

Bit 0—External Interrupt Enable
Bit 1—Timer Interrupt Enable
Bit 2—EXT INT Active Level
Bit 3—Start/Stop Timer
Bit 4—Pulse Width/Interval Timer
Bit 5— ÷ 2 Timer Prescale Values
Bit 6— ÷ 5 Timer Prescale Values
Bit 7— ÷ 20 Timer Prescale Values

A special situation exists when reading the ICP with an IN or INS instruction. The accumulator is not loaded with

Table 2 Timer Characteristics

Characteristic	Value
Interval Timer Mode	
Single Interval Error, Free-Running (Note 3)	$\pm 6t\phi$
Cumulative Interval Error, Free-Running (Note 3)	0
Error Between Two Timer Reads (Note 2)	$\pm (tpsc + t\phi)$
Start Timer to Stop Timer Error (Notes 1, 4)	$+t\phi$ to $-(tpsc + t\phi)$
Start Timer to Read Timer Error (Notes 1, 2)	$-5t\phi$ to $-(tpsc + 7t\phi)$
Start Timer to Interrupt Request Error (Notes 1, 3)	$-2t\phi$ to $-8t\phi$
Load Timer to Stop Timer Error (Note 1)	$+t\phi$ to $-(tpsc + 2t\phi)$
Load Timer to Read Timer Error (Notes 1, 2)	$-5t\phi$ to $-(tpsc + 8t\phi)$
Load Timer to Interrupt Request Error (Notes 1, 3)	$-2t\phi$ to $-9t\phi$
Pulse Width Measurement Mode	
Measurement Accuracy (Note 4)	$+t\phi$ to $-(tpsc + 2t\phi)$
Minimum Pulse Width of EXT INT Pin	$2t\phi$
Event Counter Mode	
Minimum Active Time of EXT INT Pin	$2t\phi$
Minimum Inactive Time of EXT INT Pin	$2t\phi$

Definitions

Error = indicated time value – actual time value
tpsc = $t\phi$ × prescale value

Notes

1. All times that entail loading, starting, or stopping the timer are referenced from the end of the last machine cycle of the OUT or OUTS instruction.
2. All times that entail reading the timer are referenced from the end of the last machine cycle of the IN or INS instruction.
3. All times that entail the generation of an interrupt request are referenced from the start of the machine cycle in which the appropriate interrupt request latch is set. Additional time may elapse if the interrupt request occurs during a privileged or multi-cycle instruction.
4. Error may be cumulative if operation is repetitively performed.

the contents of the ICP; instead, accumulator bits 0 through 6 are loaded with zeros, while bit 7 is loaded with the logic level being applied to the EXT INT pin, thus allowing the status of the EXT INT pin to be determined without the necessity of servicing an external interrupt request. This capability is useful in establishing a high-speed, polled handshake procedure or for using EXT INT as an extra input pin if external interrupts are not required and the timer is used only in the interval timer mode.

The rate at which the timer is clocked in the interval timer mode is determined by the frequency of an internal ϕ clock and by the division value selected for the prescaler. (The internal clock operates at one-half the external time base frequency.) If ICP bit 5 is set and bits 6 and 7 are cleared, the prescaler divides ϕ by 2. Likewise, if bit 6 or 7 is individually set, the prescaler divides ϕ by 5 or 20, respectively. Combinations of bits 5, 6, and 7 may also be selected. For example, if bits 5 and 7 are set while bit 6 is cleared, the prescaler divides by 40. Thus, possible prescaler values are: $\div 2$, $\div 5$, $\div 10$, $\div 20$, $\div 40$, $\div 100$, and $\div 200$.

Any of three conditions cause the prescaler to be reset: whenever the timer is stopped by clearing ICP bit 3, on execution of an output instruction to port 7 (the timer is assigned port address 7), or on the trailing edge transition of the EXT INT pin when in the pulse width measurement mode. These last two conditions are explained in the following paragraphs.

An OUT or OUTS instruction to port 7 loads the contents of the accumulator into both the timer and the 8-bit modulo-N register, resets the prescaler, and clears any previously stored timer interrupt request. As previously noted, the timer is an 8-bit down counter that is clocked by the prescaler in the interval timer mode and in the pulse width measurement mode. The prescaler is not used in the event counter mode. The modulo-N register is a buffer whose function is to save the value that was most recently output to port 7. The modulo-N register is used in all three timer modes.

Interval Timer Mode — When ICP bit 4 is cleared (logic 0) and at least one prescale bit is set, the timer operates in the interval timer mode. When bit 3 of the ICP is set, the timer starts counting down from the modulo-N value. After counting down to H'01', the timer returns to the modulo-N value at the next count. On the transition from H'01' to H'N', the timer sets a timer interrupt request latch. Note that the interrupt request latch is set by the transition of H'N' in the timer, thus allowing a full 256 counts if the modulo-N register is preset to H'00'. If bit 1 of the ICP is set, the interrupt request is passed to the CPU section of the F3872. However, if bit 1 of the ICP is a logic 0, the interrupt request is not passed, but the interrupt request latch remains set. If ICP bit 1 is subsequently set, the interrupt request is then passed to the CPU. Only two events can reset the timer interrupt request latch: when the timer interrupt request is acknowledged by the CPU, or when a new load of the modulo-N register is performed.

Fig. 4 Timer and Interrupt Control Port Block Diagram

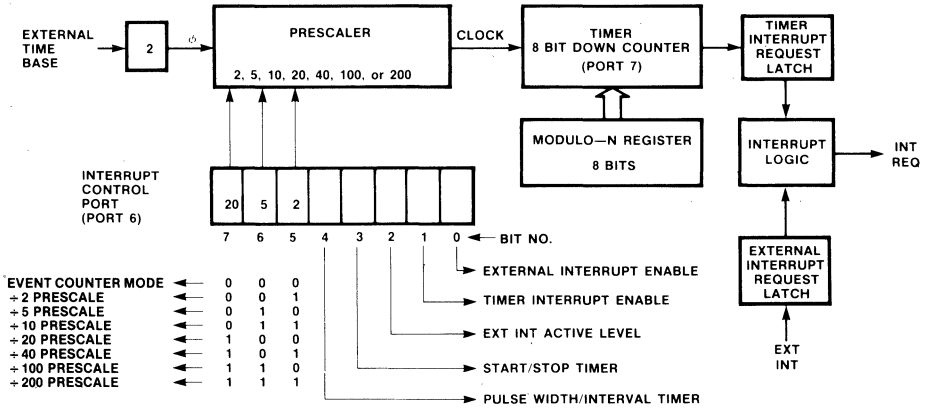
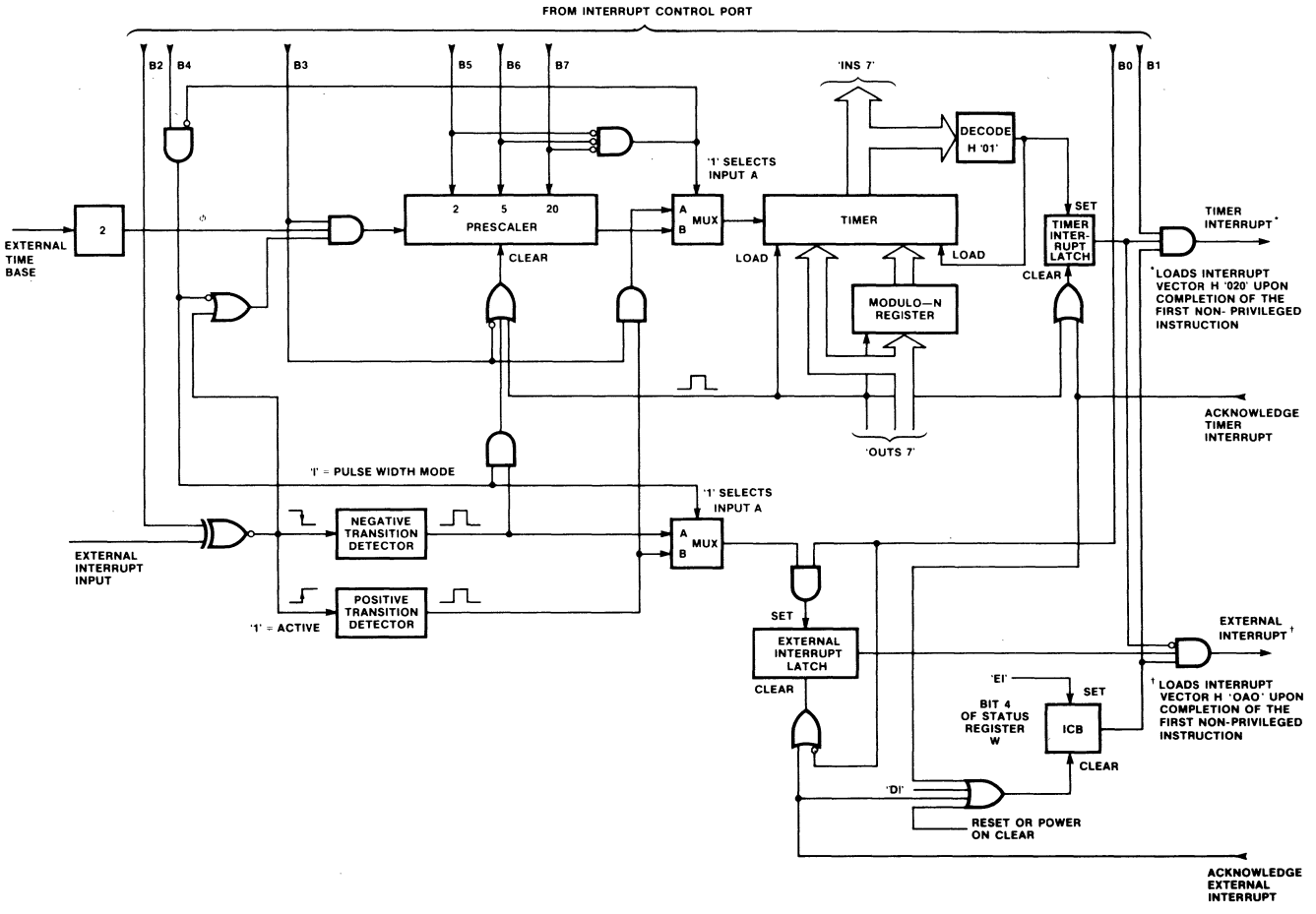


Fig. 5 Timer/Interrupt Functional Diagram



Consider an example in which the modulo-N register is loaded with H'64' (decimal 100). The timer interrupt request latch is set at the 100th count following the timer start, and the timer interrupt request latch is repeatedly set on precise 100-count intervals. If the prescaler is set at ÷ 40, the timer interrupt request latch is set every 4000 ϕ clock periods. For a 2 MHz ϕ clock (4 MHz time base frequency), this produces 2 ms intervals.

The range of possible intervals is from 2 to 51,200 ϕ clock periods (1 μs to 25.6 ms for a 2 MHz clock). However, approximately 50 ϕ periods is a practical minimum because the time between setting the interrupt request latch and the execution of the first instruction of the interrupt service routine is at least 29 ϕ periods (the response time is dependent upon how many privileged instructions are encountered when the request occurs). To establish time intervals greater than 51,200 clock periods is simply a matter of using the timer interrupt service routine to count the number of interrupts, saving the result in one or more of the scratchpad registers until the desired interval is achieved. With this technique, virtually any time interval, or several time intervals, may be generated.

The timer may be read at any time and in any mode using an input instruction (IN 7 or INS 7); this may take place on-the-fly without interfering with normal timer operation. The timer may also be stopped at any time by clearing bit 3 of the ICP. The timer holds its current contents indefinitely and resumes counting when bit 3 is again set. The prescaler, however, is reset whenever the timer is stopped; thus, a series of starts and stops results in a cumulative truncation error.

For a free-running timer in the interval timer mode, the time interval between any two interrupt requests may be in error by ± 6 ϕ clock periods, although the cumulative error over many intervals is zero. The prescaler and timer generate precise intervals for setting the timer interrupt request latch, but the time-out may occur at any time within a machine cycle. (There are two types of machine cycles: short cycles that consist of four ϕ clock periods, and long cycles that consist of 6 ϕ clock periods.) In the multi-chip F8 family, there is a signal referred to as the write clock, which corresponds to a machine cycle. Interrupt requests are synchronized with the internal write clock, thus giving rise to the possible ± 6 ϕ error. Additional errors may arise due to the interrupt request occurring while a privileged instruction or multi-cycle instruction is being executed. Nevertheless, for most applications, all of the above errors are negligible, especially if the desired time interval is greater than 1 ms.

Pulse Width Measurement Mode — When ICP bit 4 is set (logic 1) and at least one prescale bit is set, the timer operates in the pulse width measurement mode. This mode is used for accurately measuring the duration of a pulse applied to the EXT INT pin. The timer is stopped and the prescaler is reset when the EXT INT pin is at its inactive level. The active level of EXT INT is defined by ICP bit 2: if cleared, EXT INT is active-low; if set, EXT INT is active-high. If ICP bit 3 is set, the prescaler and timer start counting when EXT INT transfers to the active level. When EXT INT returns to the inactive level, the timer stops, the prescaler resets, and, if ICP bit 0 is set, an external interrupt request latch is set. (Unlike timer interrupts, external interrupts are not latched if the ICP interrupt enable bit is not set.)

As in the interval timer mode, the timer may be read at any time, or may be stopped at any time by clearing ICP bit 3, the prescaler and the ICP bit 1 function as previously described; the timer still functions as an 8-bit binary down counter with the timer interrupt request latch being set on the timer's transition from H'01' to H'N' (modulo-N value). Note that the EXT INT pin has nothing to do with loading the timer; its action is that of automatically starting and stopping the timer and of generating external interrupts. Pulse widths longer than the prescaler value times the modulo-N value are easily measured by using the timer interrupt service routine to store the number of timer interrupts in one or more scratchpad registers.

As for accuracy, the actual pulse duration is typically slightly longer than the measured value because the status of the prescaler is not readable and is reset when the timer is stopped. Thus, for maximum accuracy, it is advisable to use a small-division setting for the prescaler.

Event Counter Mode — When ICP bit 4 is cleared and all prescale bits (ICP bits 5, 6, and 7) are cleared, the timer operates in the event counter mode. This mode is used for counting pulses applied to the EXT INT pin. If ICP bit 3 is set, the timer decrements on each transition from the inactive level to the active level of the EXT INT pin. The prescaler is not used in this mode but, as in the other two timer modes, the timer may be read at any time, or may be stopped at any time by clearing ICP bit 3; ICP bit 1 functions are previously described, and the timer interrupt request latch is set on the timer's transition from H'01' to H'N' (modulo-N value).

Normally, ICP bit 0 should be kept cleared in the event counter mode; otherwise, external interrupts are generated on the transition from the inactive level to the active level of the EXT INT pin.

For the event counter mode, the minimum pulse width required on the EXT INT pin is 2 ϕ clock periods, and the minimum inactive time is 2 ϕ clock periods; therefore, the maximum repetition rate is 500 Hz.

External Interrupts
When the timer is in the interval timer mode, the EXT INT pin is available for non-timer-related interrupts. If ICP bit 0 is set, an external interrupt request latch is set when there is a transition from the inactive level to the active level of the EXT INT pin (EXT INT is an edge-triggered input). The interrupt request is latched until either acknowledged by the CPU or ICP bit 0 is cleared (unlike timer interrupt requests, which remain latched even when ICP bit 1 is cleared). External interrupts are handled in the same fashion when the timer is in the pulse width measurement mode or in the event counter mode, except that in the pulse width measurement mode the external interrupt request latch is set on the trailing edge of the EXT INT input; that is, on the transition from the active level to the inactive level.

Interrupt Handling
When either a timer or an external interrupt request is communicated to the CPU section of the F3872, it is acknowledged and processed at the completion of the first non-privileged instruction if the interrupt control bit of the status register is set. If the interrupt control bit is not set, the interrupt request continues either until the interrupt control bit is set and the CPU acknowledges the interrupt or until the interrupt request is cleared as previously described.

If there are a timer interrupt request and an external interrupt request when the CPU starts to process the requests, the timer interrupt is handled first.

When an interrupt is allowed, the CPU requests that the interrupting element pass its interrupt vector address to the program counter via the data bus. The vector address for a timer interrupt is H'20'; the vector address for an external interrupt is H'0A0'. After the vector address is passed to the program counter, the CPU sends an acknowledge signal to the appropriate interrupt request latch, which clears that latch. The execution of the interrupt service routine then commences. The return address of the original program is automatically saved in the stack register, P.

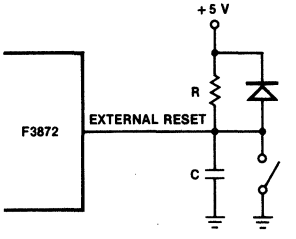
Power-On Clear
The F3872 contains power-on clear circuitry to automatically reset the internal logic following the application of external power. Since many variations of power supply circuitry exist, Fairchild cannot guarantee that the power-on clear will operate under every power-up condition.

The power-on clear circuitry contains on-chip sensors to monitor various conditions. The following conditions must be satisfied before the power-reset sequence is allowed to start:

1. Supply voltage must be above a certain value, typically +3 V to +4 V.
2. The clocks of the device must be functioning.
3. The substrate bias must reach a certain level.

All three conditions must be met before the power-on clear circuitry initiates a reset cycle. However, these conditions can be satisfied even with a supply voltage of as low as 3 volts. The latest versions of the F3872 have a modified delay circuit that gives a typical delay of 500 μs (with a 4 MHz crystal) after the above conditions are met. This is an improvement over the earlier F3872 versions.

Since the F3872 is only guaranteed to operate at a supply voltage of 4.5 V or greater, the user must ensure that the supply voltage is at least 4.5 V when the F3872 initiates the reset cycle. For power supplies having a slow rise time, an external RC network can be converted to the external reset input of the F3872 to hold the device in a reset state long enough to allow the power supply to reach a voltage of 4.5 V.



External Reset
When the $\overline{\text{RESET}}$ input is low, the contents of the program counter are pushed to the stack register and the program counter and the ICB of the status register are cleared. The original stack register contents are lost. As with power-on clear, ports 4, 5, 6, and 7 are loaded with H'00'. The contents of all other registers and ports are unchanged. When $\overline{\text{RESET}}$ is high, the first program instruction is fetched from ROM location H'0000'.

Test Logic
Special test logic is implemented to allow access to the internal main data bus for test purposes.

In normal operation, the TEST pin is unconnected or is connected to ground. When TEST is placed at a level of from 2.8 V to 3.0 V, port 4 becomes an output of the internal data bus and port 5 becomes a wired-OR input to the internal data bus. The data appearing on the port 4 pins is logically true, whereas input data forced on port 5 must be logically false. When TEST is placed at a high level (8.8 V to 9.0 V), the ports act as described above and, additionally, the program ROM is prevented from driving the data bus. In this mode, operands and instructions may be forced externally through port 5 instead of being accessed from the program ROM. When TEST is in either the TTL state or the high state, $\overline{\text{STROBE}}$ ceases its normal function and becomes a cycle clock (identical to the F8 multi-chip system write clock, except inverted).

Timing complexities render the capabilities associated with the TEST pin impractical for use in a user application, but these capabilities are sufficient to enable Fairchild to implement a rapid method for thoroughly testing the F3872.

Standby Power Option

If the standby power option is not selected, bits 0 and 1 of port 0 can be read from and written to. If the standby power-down option is selected, port 0 bit 1 is readable only; bit 0 remains both readable and writeable via software, although it is not connected to a package lead. The standby power source (V_{SB}) is connected to pin 4. (A 0.01 μF capacitor must be connected to pin 3; the purpose of this capacitor is to decouple noise coupled to the substrate of the circuit when V_{DD} is switched off and on.) Nickel-cadmium batteries (typical voltage of three series cells is 3.6 V) are recommended for use as the standby power source, since the F3872 can automatically trickle charge three such cells. If more than three cells in series are used, a charging circuit must be provided outside the F3872. When the $\overline{\text{RESET/RAMPRT}}$ pin is brought low, the standby RAM (64 8-bit words in P0/DC address spaces 4032 to 4096_{10}, or $FC0_{16}$ to FFF_{16}) is disabled from being read from or written to. The RAM itself is also switched from V_{DD} power to the V_{SB} power.

Two modes are recommended for powering down. In the first mode (see *Figure 6A*), the processor must be interrupted early enough to save all necessary data before the V_{CC} falls below the minimum level. After the save is done, the $\overline{\text{RESET/RAMPRT}}$ pin can fall. This prevents any further RAM accesses; V_{DD} may then fall.

The second mode (see *Figure 6B*) may be used if a special save data routine is not needed. External interrupt need not be used, and the only requirement to save the RAM data is that $\overline{\text{RAMPRT}}$ be low for V_{DD} drops below 4.5 V. For example, if a few key variables are to be stored in power-down RAM and it is desired that these be saved during a loss of power, two copies of each variable are kept with an associated flag in the power-down RAM; thus, no interrupt and save routine is necessary. The method of updating a variable is as follows:

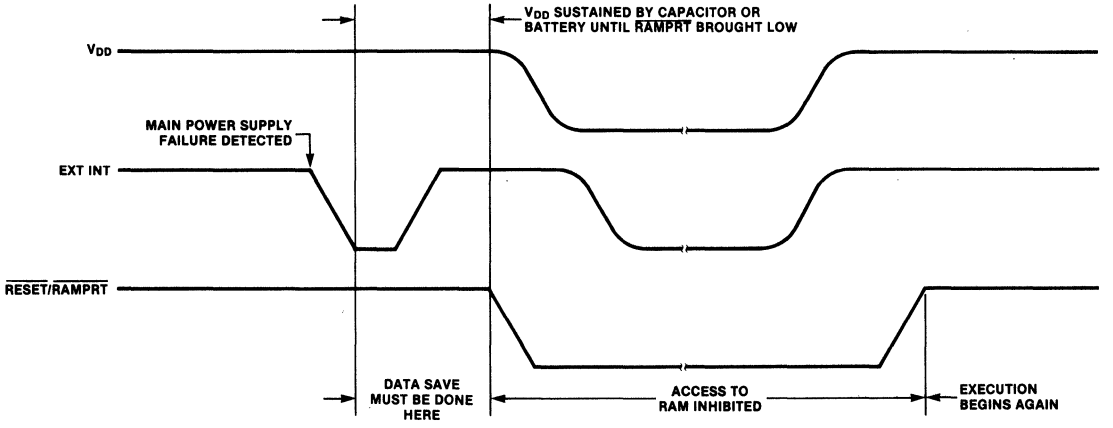
Clear Flag Word 1
Update Variable (Copy 1)
Set Flag Word 1
Clear Flag Word 2
Update Variable (Copy 2)
Set Flag Word 2

Execution may terminate at any time, even during the update of a variable of flag word, causing that byte in scratchpad to be "bad" data. There is always a "good" data byte that contains either the most recent or next-most recent value of the variable. Any copy of the variable in which the flag word is set is a good data byte While this method significantly encumbers the data storage process, it eliminates the need for a power fail interrupt, which reduces external circuitry and leaves the external interrupt pin completely free for other uses.

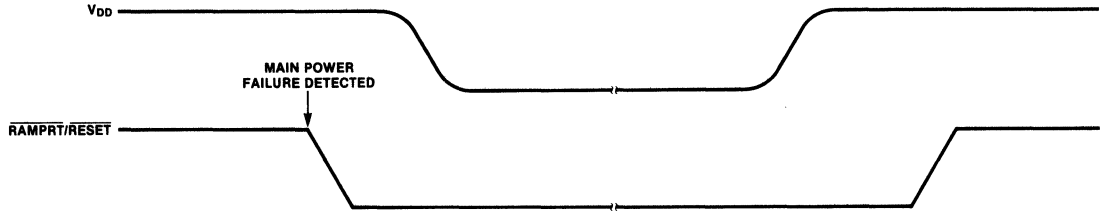
In either power-down mode, the $\overline{\text{RESET/RAMPRT}}$ signal should be held low until V_{DD} is above the minimum level when power returns.

Fig. 6 Standby Power Option Modes

A. Data Save Routine, $V_{SB} \leq 3.2$ V



B. No Save Routine, $V_{SB} \leq 3.2$ V



F3872 Clocks

The time bases for the F3872 may originate from one of four external sources; the four external configurations are shown in *Figure 7.* There is an internal 26.5 pF capacitor between XTL_1 and GND, and also between XTL_2 and GND. Thus, external capacitors are not required. In all external clock modes, the external time base frequency is divided by 2 to form the internal ϕ clock.

Instruction Set

The F3872 executes the entire instruction set of the multi-chip F8 family (F3850 family), as shown in *Table 3.* Of course, the STORE instruction only accesses memory in locations FF0–FFF (the data counter, however, is incremented each time STORE is executed).

A summary of programmable registers and ports is given in *Figure 8.*

Also, for convenient reference, a programming model of the F3872 is given in *Figure 9.*

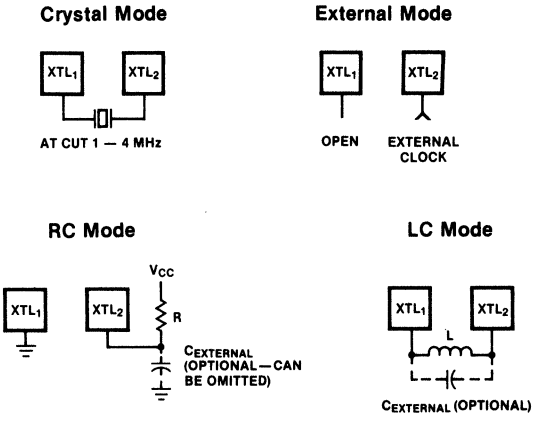
Mask Options

The ROM array may contain object program code and/or tables of nonvarying data. Every F3872 is implemented using a custom mask that specifies the state of every ROM bit, as well as certain address mask options that are external to the ROM array. The following mask options are specified:

1. The 1024, 2048, 3072, or 4096 bytes of ROM storage. This reflects programs and permanent data tables stored in the PSU memory.
2. Input/output ports can be any of the following three configurations:
   a. Standard pull-up
   b. Open drain
   c. Direct drive
3. Input/output ports 0 and 1 can be specified either cleared or unaltered following an external reset.
4. External interrupt and external reset can be specified to have or omit an internal pull-up resistor.
5. The I/O port output option choices are: the standard pull-up (option A), the open drain (option B), and the driver pull-up (option C).

The format for mask options must be submitted to Fairchild Microprocessor Division before device manufacture. The data to be stored in permanent memory may be submitted in the form of an EPROM or HP2644/HP2645 cartridge (Formulator format only). Other options must be specified on the Fairchild ROM Code Entry Form, available from a Fairchild representative.

Fig. 7 F3872 Clock Configurations



Supplementary Notes

For total software compatibility when expanding into a multi-chip configuration, the F3871 Peripheral Input/Output circuit should be used. The F3871 has the same improved timer (binary count, readable, and three modes of operation) and ready strobe outputs as the F3872.

The interrupt control bit of the status register is automatically reset when an interrupt request is acknowledged. It is then the programmer's responsibility to determine when the ICB is again to be set (by executing the E1 instruction). This action prevents an interrupt service routine from being interrupted unless the programmer so desires.

Table 3 F3872 Instruction Set

Accumulator Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Carry	LNK		ACC←(ACC)+CRY	19	1	1	1/0	1/0	1/0	1/0
Add Immediate	AI	ii	ACC←(ACC)+H'ii'	24 ii	2	2.5	1/0	1/0	1/0	1/0
AND Immediate	NI	ii	ACC←(ACC) Λ H'ii'	21 ii	2	2.5	0	1/0	0	1/0
Clear	CLR		ACC←H'00'	70	1	1				
Compare Immediate	CI	ii	H'ii'+($\overline{ACC}$)+1	25 ii	2	2.5	1/0	1/0	1/0	1/0
Complement	COM		ACC←(ACC)⊕H'FF'	18	1	1	0	1/0	0	1/0
Exclusive OR Immediate	XI	ii	ACC←(ACC)⊕H'ii'	23 ii	2	2.5	0	1/0	0	1/0
Increment	INC		ACC←(ACC)+1	1F	1	1	1/0	1/0	1/0	1/0
Load Immediate	LI	ii	ACC←H'ii'	20 ii	2	2.5	—	—	—	—
Load Immediate Short	LIS	i	ACC←H'0i'	7 i	1	1	—	—	—	—
OR Immediate	OI	ii	ACC←(ACC) V H'ii'	22 ii	2	2.5	0	1/0	0	1/0
Shift Left One	SL	1	SHIFT LEFT 1	13	1	1	0	1/0	0	1/0
Shift Left Four	SL	4	SHIFT LEFT 4	15	1	1	0	1/0	0	1/0
Shift Right One	SR	1	SHIFT RIGHT 1	12	1	1	0	1/0	0	1
Shift Right Four	SR	4	SHIFT RIGHT 4	14	1	1	0	1/0	0	1

Branch Instructions

(In All Conditional Branches, P0 (P0)+2 if the Test Conditions Are Not Met. Execution Is Complete in 30 Cycles.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Branch on Carry	BC	aa	P0←[(P0)+1]+H'aa' if CRY=1	82 aa	2	3.5	—	—	—	—
Branch on Positive	BP	aa	P0←[(P0)+1]+H'aa' if SIGN=1	81 aa	2	3.5	—	—	—	—
Branch on Zero	BZ	aa	P0←[(P0)+1]+H'aa' if ZERO=1	84 aa	2	3.5	—	—	—	—
Branch on True	BT	t,aa	P0←[(P0)+1]+H'aa' if any test is true	8t aa	2	3.5	—	—	—	—
			t=TEST CONDITION							
			2^2 | 2^1 | 2^0							
			ZERO | CRY | SIGN							
Branch if Negative	BM	aa	P0←[(P0)+1]+H'aa' if SIGN=0	91 aa	2	3.5	—	—	—	—
Branch if No Carry	BNC	aa	P0←[(P0)+1]+H'aa' if CARRY≠0	92 aa	2	3.5	—	—	—	—
Branch if No Overflow	BNO	aa	P0←[(P0)+1]+H'aa' if OVF=0	98 aa	2	3.5	—	—	—	—
Branch if Not Zero	BNZ	aa	P0←[(P0)+1]+H'aa' if ZERO=0	94 aa	2	3.5	—	—	—	—
Branch if False Test	BF	t,aa	P0←[(P0)+1]+H'aa' if all false test bits	9t aa	2	3.5	—	—	—	—
			t=TEST CONDITION							
			2^3 | 2^2 | 2^1 | 2^0							
			OVF | ZERO | CRY | SIGN							
Branch if ISAR (Lower) 7	BR7	aa	P0←[(P0)+1]+H'aa' if ISARL≠7	8F aa	2	2.5	—	—	—	—
			P0←(P0)+2 if ISARL=7			2.0	—	—	—	—
Branch Relative	BR	aa	P0←[(P0)+1]+H'aa'	90 aa	2	3.5	—	—	—	—
Jump*	JMP	aaaa	P0←H'aaaa'	29 aaaa	3	5.5	—	—	—	—

Memory Reference Instructions (In All Memory Reference Instructions, the Data Counter Is Incremented DC←DC•1.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Binary	AM		ACC←(ACC)+[(DC)]	88	1	2.5	1/0	1/0	1/0	1/0
Add Decimal	AMD		ACC←(ACC)+[(DC)]	89	1	2.5	1/0	1/0	1/0	1/0
AND	NM		ACC←(ACC) Λ [(DC)]	8A	1	2.5	0	1/0	0	1/0
Compare	CM		[(DC)]+($\overline{ACC}$)+1	8D	1	2.5	1/0	1/0	1/0	1/0
Exclusive OR	XM		ACC←(ACC)⊕[(DC)]	8C	1	2.5	0	1/0	0	1/0
Load	LM		ACC←[(DC)]	16	1	2.5	—	—	—	—
Logical OR	OM		ACC←(ACC) V [(DC)]	8B	1	2.5	0	1/0	0	1/0
Store	ST		(DC)←(ACC)	17	1	2.5	—	—	—	—

*Privileged instruction

Note

JMP and PI change accumulator contents to the high byte address.

Table 3 F3872 Instruction Set (Cont.)

Address Register Group Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add to Data Counter	ADC		DC←(DC)+(ACC)	8E	1	2.5	—	—	—	—
Call to Subroutine	PK*		P←(P0); P0U←(r12); PL←(r13)	0C	1	4	—	—	—	—
Call to Subroutine Immediate	PI*	aaaa	P←(P); P0←H'aaaa'‡	28 aaaa	3	6.5	—	—	—	—
Exchange DC	XDC		DC—DC1	2C	1	2	—	—	—	—
Load Data Counter	LR	DC,Q	DCU←(r14); DCL←(r15)	0F	1	4	—	—	—	—
Load Data Counter	LR	DC,H	DCU←(r10); DCL←(r11)	10	1	4	—	—	—	—
Load DC Immediate	DCI	aaaa	DC←H'aaaa'	2A aaaa	3	6	—	—	—	—
Load Program Counter	LR	P0,Q	P0U←(r14); P0L←(r15)	0D	1	4	—	—	—	—
Load Stack Register	LR	P,K	PU←(r12); PL←(r13)	09	1	4	—	—	—	—
Return From Subroutine	POP*		P0↔(P)	1C	1	2	—	—	—	—
Store Data Counter	LR	Q,DC	r14←(DCU); r15←(DCL)	0E	1	4	—	—	—	—
Store Data Counter	LR	H,DC	r10←(DCU); r11←(DCL)	11	1	4	—	—	—	—
Store Stack Register	LR	K,P	r12←(PU); r13←(P)	08	1	4	—	—	—	—

Scratchpad Register Instructions (Refer to Scratchpad Addressing Modes.)

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Add Binary	AS	r	ACC←(ACC)+(r)	Cr	1	1	1/0	1/0	1/0	1/0
Add Decimal	ASD	r	ACC←(ACC)+(r)	Dr	1	2	1/0	1/0	1/0	1/0
Decrement	DS	r	r←(r)+H'FF'	3r	1	1.5	1/0	1/0	1/0	1/0
Load	LR	A,r	ACC←(r)	4r	1	1	—	—	—	—
Load	LR	A,KU	ACC←(r12)	00	1	1	—	—	—	—
Load	LR	A,KL	ACC←(r13)	01	1	1	—	—	—	—
Load	LR	A,QU	ACC←(r14)	02	1	1	—	—	—	—
Load	LR	A,QL	ACC←(r15)	03	1	1	—	—	—	—
Load	LR	r,A	r←(ACC)	5r	1	1	—	—	—	—
Load	LR	KU,A	r12←(ACC)	04	1	1	—	—	—	—
Load	LR	KL,A	r13←(ACC)	05	1	1	—	—	—	—
Load	LR	QU,A	r14←(ACC)	06	1	1	—	—	—	—
Load	LR	QL,A	r15←(ACC)	07	1	1	—	—	—	—
AND	NS	r	ACC←(ACC) ∧ (r)	Fr	1	1	0	1/0	0	1/0
Exclusive OR	XS	r	ACC←(ACC)⊕(r)	Er	1	1	0	1/0	0	1/0

Miscellaneous Instructions

Operation	Mnemonic OP Code	Operand	Function	Machine Code	Bytes	Cycles	Status Bits OVF	ZERO	CRY	SIGN
Disable Interrupt	DI		RESET ICB	1A	1	2	—	—	—	—
Enable Interrupt*	EI		SET ICB	1B	1	2	—	—	—	—
Input	IN	aa	ACC←(INPUT PORT aa)	26 aa	2	4	0	1/0	0	1/0
Input Short	INS	a	ACC←(INPUT PORT a)	Aa	1	4***	0	1/0	0	1/0
Load ISAR	LR	IS,A	ISAR←(ACC)	0B	1	1	—	—	—	—
Load ISAR Lower	LISL	a	ISARL←a	01101a**	1	1	—	—	—	—
Load ISAR Upper	LISU	a	ISARU←a	01100a**	1	1	—	—	—	—
Load Status Register*	LR	W,J	W←(r9)	1D	1	2	1/0	1/0	1/0	1/0
No-Operation	NOP		P0←(P0)+1	2B	1	1	—	—	—	—
Output	OUT	aa	OUTPUT PORT aa←(ACC)	27 aa	2	4	—	—	—	—
Output Short	OUTS	a	OUTPUT PORT a←(ACC)	Ba	1	4***	—	—	—	—
Store ISAR	LR	A,IS	ACC←(ISAR)	0A	1	1	—	—	—	—
Store Status Register	LR	J,W	r9←(W)	1E	1	1	—	—	—	—

*Privileged instruction
**3-bit octal digit
***Two machine cycles for CPU ports
‡Contents of ACC destroyed

Table 3 F3872 Instruction Set (Cont.)

Notes
Each lower case character represents a hexadecimal digit.
Each cycle equals four machine clock periods.
Lower case denotes variables specified by the programmer.

Function Definitons

←	is replaced by
( )	the contents of
(—)	binary ones complement of
+	arithmetic add (binary or decimal)
⊕	logical OR exclusive
Λ	logical AND
V	logical OR inclusive
H'#'	hexadecimal digit

Register Names

a	address variable
A	accumulator
DC	data counter (indirect address register)
DC1	data counter #1 (auxiliary data counter)
DCL	least significant eight bits of data counter addressed
DCU	most significant eight bits of data counter addressed
H	scratchpad register #10 and #11
i and ii	immediate operand
ICB	interrupt control bit
IS	indirect scratchpad address register
ISAR	indirect scratchpad address register
ISARL	least significant three bits of ISAR
ISARU	most significant three bits of ISAR
J	scratchpad register #9
K	registers #12 and #13
KL	register #13
KU	register #12
P0	program counter
P0L	least significant eight bits of program counter
P0U	most significant eight bits of program counter
P	stack register
PL	least significant eight bits of program counter
PU	most significant eight bits of active stack register
Q	registers #14 and #15
QL	register #15
QU	register #14
r	scratchpad register (any address through 11)
W	status register

Scratchpad Addressing Modes (Machine Code Format)

r = C	(hexadecimal) register addressed by ISAR (unmodified)
r = D	(hexadecimal) register addressed by ISAR; ISARL incremented
r = E	(hexadecimal) register addressed by ISAR; ISARL decremented
r = F	(no operation performed)
r = 0-B	(hexadecimal) register 0 through 11 addressed directly from the instruction

Status Register

—	no change in condition
1/0	is set to 1 or 0, depending on conditions
CRY	carry flag

When reading the interrupt control port (port 6), bit 7 of the accumulator is loaded with the actual logic level being applied to the EXT INT pin, regardless of the status of ICP bit 2 (the EXT INT active level bit); that is, if the EXT INT pin is at +5 V, bit 7 of the accumulator is set to a logic 1, but if the EXT INT pin is at ground, accumulator bit 7 is reset to logic 0.

In *Table 3,* the number of cycles shown is "nominal machine cycles." A nominal machine cycle is defined as 4 ϕ clock periods, thus requiring 2 μs for a 2 MHz clock frequency (4 MHz external time base frequency).

Table 3 also uses the following nomenclature for register names:

F8 ⟷ F3872
PC_0 = P0 Program Counter
PC_1 = P Stack Register
DC_0 = DC Data Counter
DC_1 = DC1 Auxiliary Data Counter

This nomenclature is used to be consistent with the assembly language mnemonics.

For the F3872, execution of an INS or OUTS instruction requires two machine cycles for ports 0 and 1, whereas ports 4 and 5 require four machine cycles.

When an external reset of the F3872 occurs, P0 pushes into P and the old contents of P are lost. It must be noted that an external reset is recognized at the start of the machine cycle and not necessarily at the end of an instruction. Thus, if the F3872 is executing a multi-cycle instruction, that instruction is not completed and the contents of P upon reset may not necessarily be the address of the instruction that would have been executed next. It may, for example, point to an immediate operand if the reset occurred during the second cycle of an L1 or C1 instruction. Additionally, several instructions (JMP, P1, PK, LR, P0 and Q) as well as the interrupt acknowledge sequence modify P0 in parts. That is, they alter P0 by loading first one part, then the other, and the entire operation takes more than one cycle. Should reset occur during this modification process, the value pushed into P is part of the old P0 (the as-yet unmodified part) and part of the new P0 (already-modified part). Thus, care should be taken (perhaps by external gating) to ensure that reset does not occur at an undesirable time if any significance is to be given to the contents of P after a reset occurs.

Fig. 8 Programmable Registers and Ports

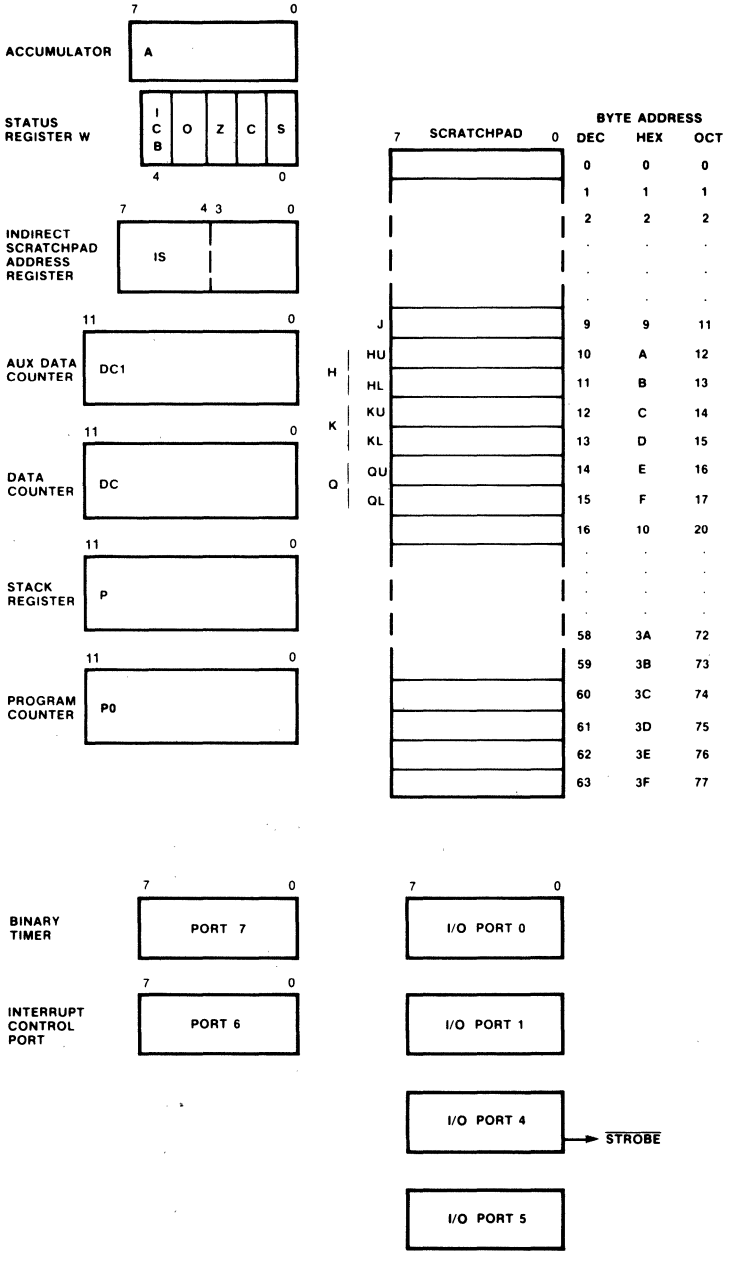
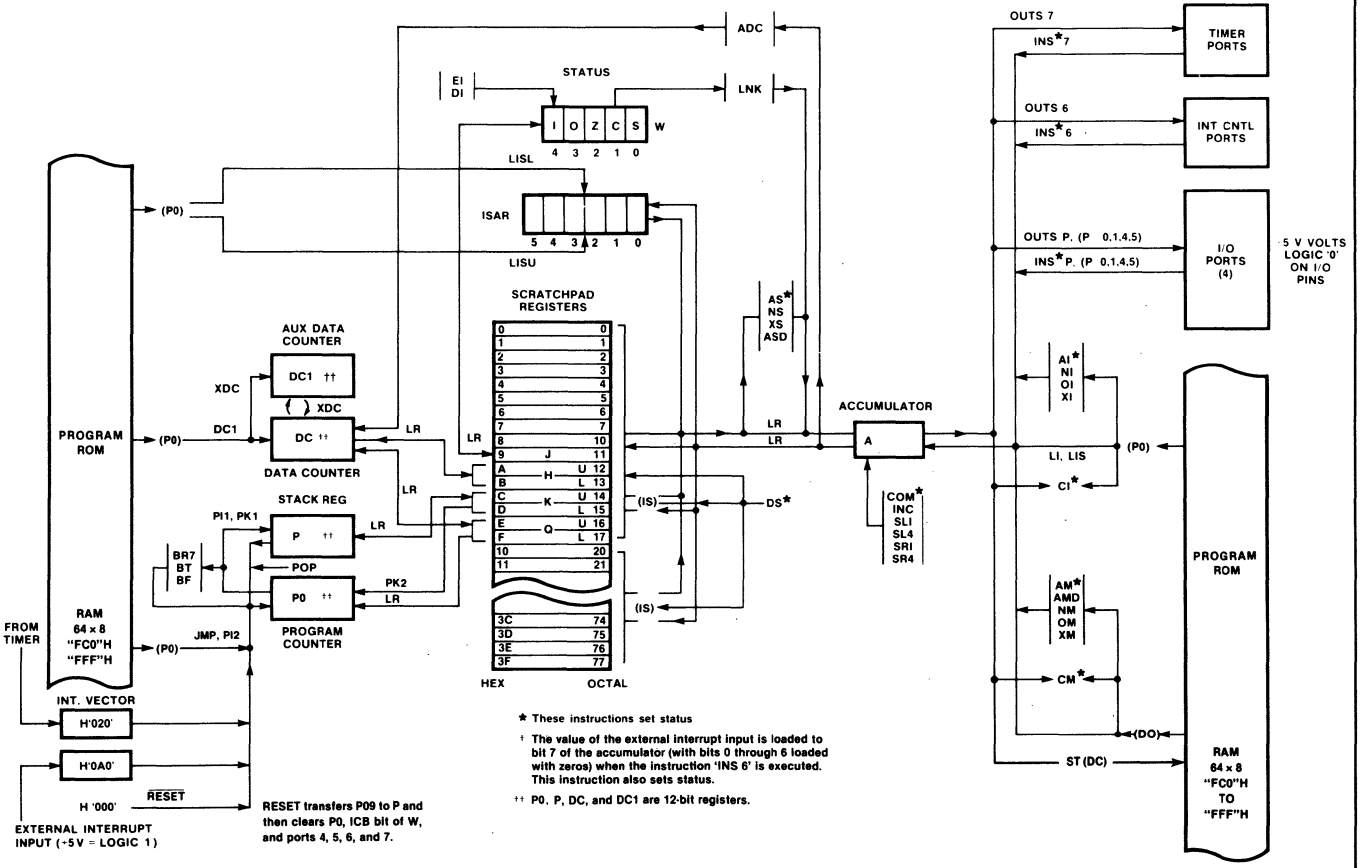


Fig. 9 Programming Model



★ These instructions set status

† The value of the external interrupt input is loaded to bit 7 of the accumulator (with bits 0 through 6 loaded with zeros) when the instruction 'INS 6' is executed. This instruction also sets status.

†† P0, P, DC, and DC1 are 12-bit registers.

Note:
The instructions PI and PK are shown in two sequential parts (PI1, PI2, and PK1, PK2).

Timing Characteristics

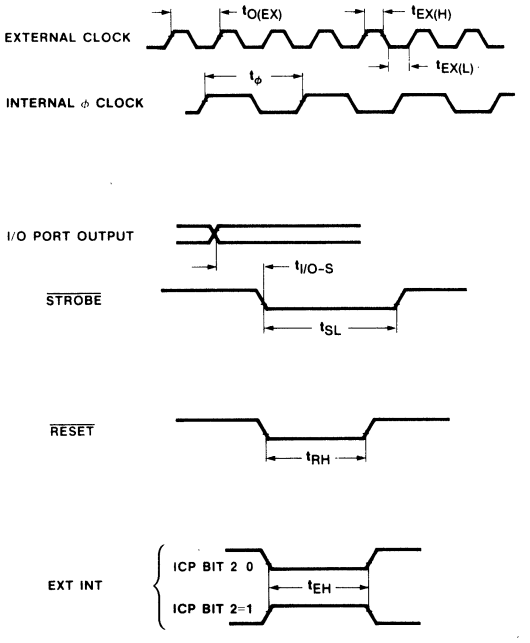
The F3872 timing characteristics are described in *Table 4* and illustrated in *Figures 10* and *11.*

Table 4 Timing Characteristics

Signal	Symbol	Characteristic	Min	Max	Unit	Notes
XTL_1 XTL_2	$t_o(EX)$	Time Base Period, All External Modes	250	1000	ns	4 MHz–1 MHz
	$t_{EX}(H)$	External Clock Pulse Width, High	90	700	ns	
	$t_{EX}(L)$	External Clock Pulse Width, Low	100	700	ns	
ϕ	t_ϕ	Internal ϕ Clock Period	$2t_\phi$			
WRITE	t_W	Internal WRITE Clock Period	$4t_\phi$ $6t_\phi$			Short Cycle Long Cycle
I/O	$t_{dI/O}$	Output Delay from Internal WRITE Clock	0	1000	ns	50 pF Plus One TTL Load
	$t_{sI/O}$	Input Setup Time to WRITE Clock	1000		ns	
$\overline{STROBE}$	$t_{I/OS}$	Output Valid to $\overline{STROBE}$ Delay	$3t_\phi$ -1000	$3t_\phi$ $+250$	ns	Note 1
	t_{SL}	$\overline{STROBE}$ Low Time	$8t_\phi$ -250	$12t_\phi$ $+250$	ns	
$\overline{RESET}$	t_{RH}	$\overline{RESET}$ Hold Time, Low	$6t_\phi$ $+750$		ns	
EXT INT	t_{EH}	EXT INT Hold Time, Active State	$6t_\phi$ $+750$		ns	To Trigger Interrupt
		EXT INT Hold Time, Inactive State	$2t_\phi$		ns	To Trigger Timer; Note 2
	C_{IN}	Input Capacitance: I/O Ports, $\overline{RESET}$, EXT INT RAMPRT, TEST		7	pF	Unmeasured Pins Returned to V_{SS}; Note 3
	C_{XTL}	Input Capacitance: XTL_1, XTL_2	23.5	29.5	pF	Unmeasured Pins Returned to V_{SS}; Note 3

Notes
1. I/O load is 50 pF plus one standard TTL input; $\overline{STROBE}$ load is 50 pF plus three standard TTL inputs.
2. Specification is applicable when the timer is in the interval timer mode.
3. $T_A = 25°C$, f = 2 MHz.
4. $T_A = 0°C$ to $+70°C$, $V_{CC} = +5\ V \pm 10\%$, I/O power dissipation $\leq$ mW, unless otherwise noted.

Fig. 10 Timing Diagrams



Note
All measurements are referenced to V_{IL} max, V_{IH} min, V_{OL} max, or V_{OH} min

4

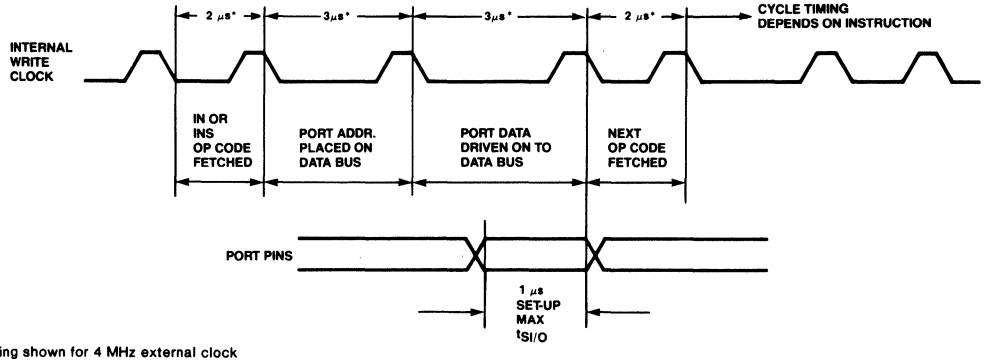
Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Voltage on any Pin with Respect to Ground (Except Open-Drain Pins)	− 1.0 V, + 7 V
Voltage on any Open-Drain Pin	− 1.0 V, + 13.2 V
Power Dissipation	1.5 W
Ambient Temperature Under Bias	0 °C, + 70 °C
Storage Temperature	− 55 °C, + 150 °C

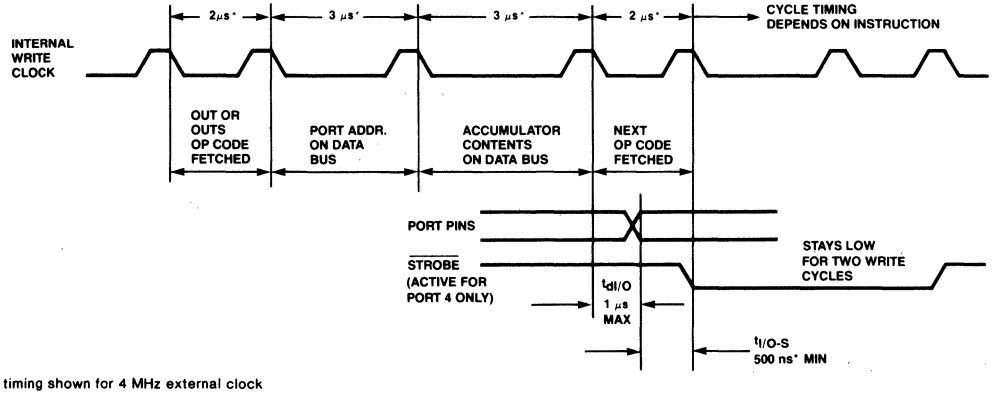
Fig. 11 Port Input/Output Timing Diagrams

A. Input on Port 4 or 5



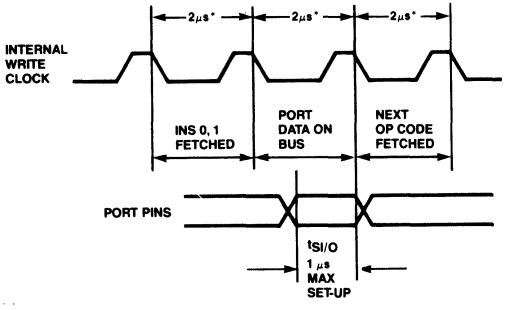
*Cycle timing shown for 4 MHz external clock

B. Output on Port 4 or 5



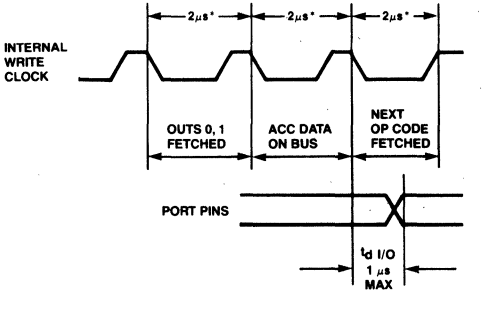
*Cycle timing shown for 4 MHz external clock

C. Input on Port 0 or 1



*Cycle timing shown for 4 MHz external clock

D. Output on Port 0 or 1



*Cycle timing shown for 4 MHz external clock

DC Characteristics

The dc characteristics of the F3872 are described in *Table 5.*

Table 5 DC Characteristics

Symbol	Characteristic	Min	Max	Unit	Conditions
I_{CC}	Power Supply Current		100	mA	Outputs Open
P_D	Power Dissipation		500	mW	Outputs Open
V_{IHEX}	External Clock Input High Voltage	2.4	5.8	V	
V_{ILHEX}	External Clock Input Low Voltage	−0.3	0.6	V	
I_{HEX}	External Clock Input High Current		100	μA	$V_{IHEX} = V_{DD}$
I_{ILEX}	External Clock Input Low Current		−100	μA	$V_{ILEX} = V_{SS}$
V_{IH}	Input High Voltage $\overline{\text{RESET}}$, EXT INT	2.0	5.8	V	$\overline{\text{RESET}}$ and EXT INT Have Internal Schmitt Triggers Giving Minimum 0.2 V Hysteresis
V_{IL}	Input Low Voltage $\overline{\text{RESET}}$, EXT INT	−0.3	0.8	V	
V_{IHOD}	Input High Voltage (Open-Drain Ports)	2.0	13.2	V	
I_{IL}	Input Low Current $\overline{\text{RESET}}$, EXT INT		−1.6	mA	$V_{IL} = 0.4V$ Note 1
I_{LOD}	Leakage Current (Open-Drain Ports)		10 −5.0	μA	$V_{IN} = 13.2$ V $V_{IN} = 0.0$ V Note 2
I_{OH}	Output High Current $\overline{\text{RESET}}$, EXT INT	−100 −30		μA	$V_{OH} = 2.4$ V $V_{OH} = 3.9$ V
I_{OHDD}	Output High Current (Direct-Drive Ports)	−0.1		mA	$V_{OH} = 2.4$ V
		−1.5		mA	$V_{OH} = 1.5$ V
			−8.5	mA	$V_{OH} = 0.7$ V
I_{OL}	Output Low Current	1.8		mA	$V_{OL} = 0.4$ V
I_{OHS}	Output High Current ($\overline{\text{STROBE}}$ Output)	−300		μA	$V_{OH} = 2.4$ V
I_{OLS}	Output Low Current ($\overline{\text{STROBE}}$ Output)	5.0		mA	$V_{OL} = 0.4$ V
V_{IHRPR}	$\overline{\text{RAMPRT}}$ Input High Level	1.9	5.8	V	Guaranteed 0.1 V less than V_{IH} for $\overline{\text{RESET}}$
V_{ILRPR}	$\overline{\text{RAMPRT}}$ Input Low Level	−0.3	0.4	V	Guaranteed 0.1 V less than V_{IL} for $\overline{\text{RESET}}$
V_{SB}	Standby V_{DD} for RAM	3.2	5.5	V	
I_{SB}	Standby Current		6.0 3.7	mA mA	$V_{SB} = 5.5$ V $V_{SB} = 3.2$ V
I_{CHG}	Trickle Charge Available on V_{SB} with V_{DD} − 4.5 to 5.5 V	−0.8	 −15	mA mA	$V_{SB} = 3.8$ V $V_{SB} = 3.2$ V
P_{DIO}	Power Dissipated by I/O Pins		600 60	mW mW	All Pins Any One Pin, Note 3

Notes
1. RESET or EXT INT programmed with standard pull-up.
2. RESET or EXT INT programmed without standard pull-up.
3. Power dissipation of I/O pins is calculated by $\Sigma\,(V_{DD} - V_{IL})\,(|I_{IL}|) + \Sigma\,(V_{DD} - V_{OH})\,(|I_{OH}|) + \Sigma\,(V_{OL})\,(I_{OL})$.
4. $T_A = 0°C$ to $+70°C$, $V_{DD} = +5$ V ± 10%, I/O power dissipation ≤ 100 mW.

Ordering Information

Order Code	Package	Temperature Range*
F3872DC, F38L72DC	Ceramic	C
F3872DL, F38L72DL	Ceramic	L
F3872DM, F38L72DM	Ceramic	M
F3872PC, F38L72PC	Plastic	C

*C = Commercial Temperature Range 0°C to +70°C
L = Limited Temperature Range −40°C to +85°C
M = Military Temperature Range −55°C to +125°C

FAIRCHILD
A Schlumberger Company

Section 5
F6800 Microprocessor Family

General

The Fairchild F6800 microprocessor family is a set of 8-bit MOS devices that offers a complete and constantly growing selection of microprocessors having a powerful instruction set. As shown in figure 5-1, the F6800 family now includes seven different CPUs (described in table 5-1), supported by such circuits as synchronous and asynchronous controllers for data communications, timers, a direct memory access controller, CRT controllers, RAMs, ROMs, and EPROMs (described in table 5-2).

Table 5-1 F6800 Microprocessor Family CPUS

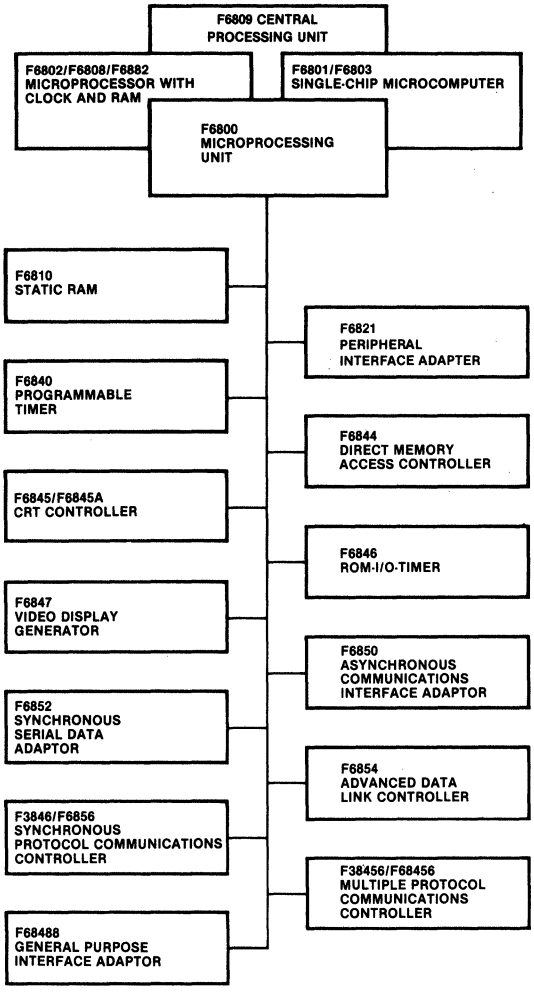
Device No.	No. of Pins	Power Supply	External Addressing	Data Length (Bits)	Clock	No. of Basic Instructions	Bytes (RAM)	Bytes (ROM)	No. of I/O Lines	Other I/O	Timer
F6800	40	+5 V	64K	8	No	—	—	—	—	—	
F6801	40	+5 V	64K	8	Yes	82	128	2K	31	Serial	16-Bit
F6802	40	+5 V	64K	8	Yes	72	128	—	—	—	—
F6803	40	+5 V	64K	8	Yes	82	128	—	13	Serial	16-Bit
F6808	40	+5 V	64K	8	Yes	72	—	—	—	—	—
F6809	40	+5 V	64K	8	Yes	59	—	—	—	—	—
F6882	40	+5 V	64K	8	Yes	72	128	—	—	—	—

Table 5-2 F6800 Peripheral Devices

Type	Number	Name	Comment
General-Purpose	F6820*	Peripheral Interface Adapter	Twenty I/O Lines
General-Purpose	F6821	Peripheral Interface Adapter	Twenty I/O Lines
General-Purpose	F6840	Programmable Timer Module	Three-to 16-Bit Timers
General-Purpose	F68488	General-Purpose Interface Adapter	IEEE-488 Bus Controller
Special Function	F6844	Direct Memory Access Controller	Three I/O Channels
Special Function	F6845	CRT Controller	Available in Interlace or Non-Interlace
Special Function	F6846	ROM, I/O, Timer	2K x 8 ROM, Parallel I/O, Timer
Special Function	F6847	Video Display Generator	Low-Cost Video Controller
Data Communications	F6850	Asynchronous Communications Interface Adapter	
Data Communications	F6852	Synchronous Serial Data Adapter	
Data Communications	F6854	Advanced Data Link Controller	HDLC/SDLC
Data Communications	F6856/ F3846	Synchronous Communications Protocol Controller	HDLC/SDLC/BTSYNC
Data Communications	F68456/ F38456	Multi-Protocol Communications Controller	HDLC/SDLC/BISYNC/ASYNC
Memory	F6810	128 x 8-Bit Static RAM	

* Not recommended for new designs.

Figure 5-1


F6809 CENTRAL PROCESSING UNIT
F6802/F6808/F6882 MICROPROCESSOR WITH CLOCK AND RAM
F6801/F6803 SINGLE-CHIP MICROCOMPUTER
F6800 MICROPROCESSING UNIT
F6810 STATIC RAM
F6821 PERIPHERAL INTERFACE ADAPTER
F6840 PROGRAMMABLE TIMER
F6844 DIRECT MEMORY ACCESS CONTROLLER
F6845/F6845A CRT CONTROLLER
F6846 ROM-I/O-TIMER
F6847 VIDEO DISPLAY GENERATOR
F6850 ASYNCHRONOUS COMMUNICATIONS INTERFACE ADAPTOR
F6852 SYNCHRONOUS SERIAL DATA ADAPTOR
F6854 ADVANCED DATA LINK CONTROLLER
F3846/F6856 SYNCHRONOUS PROTOCOL COMMUNICATIONS CONTROLLER
F38456/F68456 MULTIPLE PROTOCOL COMMUNICATIONS CONTROLLER
F68488 GENERAL PURPOSE INTERFACE ADAPTOR

Instruction Set

Because a single instruction set is inadequate for the number and flexibility of devices in the F6800 family, it has been necessary to develop three such sets, each serving a portion of the family.

The basic instruction set, comprising 72 instructions, is supported by the F6800, F6802, F6808, and F6882; figure 5-2 is the associated programming model. An expanded instruction set, consisting of the basic set plus several additional instructions, is supported by the F6801 and F6803; figure 5-3 illustrates the associated programming model. The expanded instruction set is upward-compatible with the basic set (that is, programs written using either are interchangeable, provided that the additional instructions are not involved). Both the basic and expanded instruction sets are described in table 5-3.

The instruction set supported by the high-performance F6809 is similar in structure to the basic and expanded sets, but is not upward-compatible. It is greatly enhanced to take fullest advantage of the powerful F6809 architecture. Figure 5-4 illustrates the F6809 programming model and table 5-4 describes the instruction set.

Figure 5-2 F6800/F6802/F6808/F6882 Programming Model

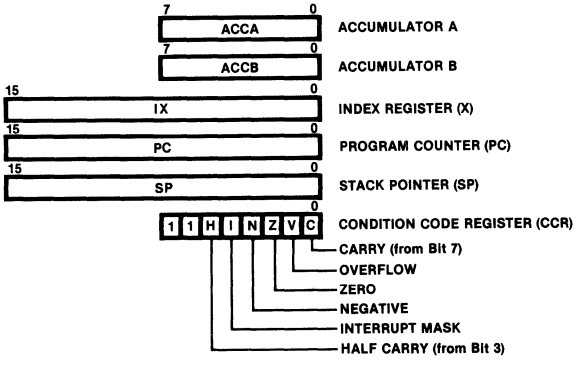


Figure 5-3 F6801/F6803 Programming Model

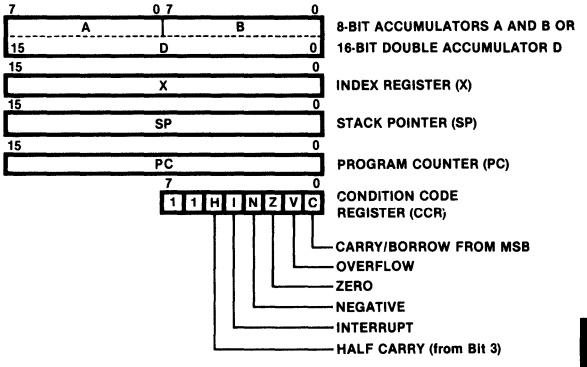


Figure 5-4 F6809 Programming Model

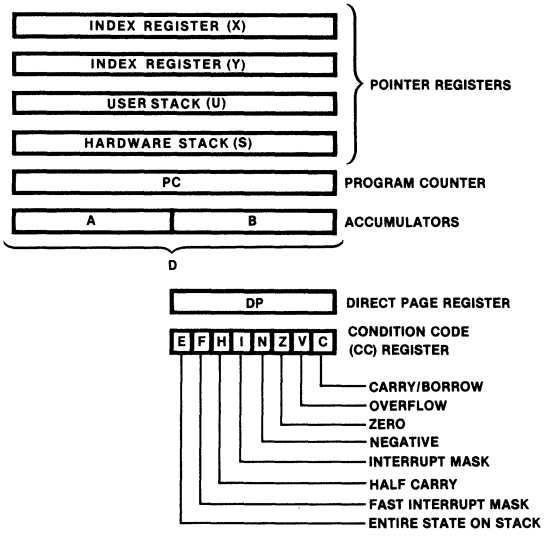


Table 5-3 Basic and Expanded Instruction Sets

Instruction	Description
ABA	Add Accumulators
*ABX	Add Accumulator B to Index Register
ADC	Add With Carry
ADD	Add
*ADDD	Add Double Accumulator to Memory; Leave Sum In Double Accumulator
AND	Logical AND
ASL	Arithmetic Shift Left
*ASLD	Double Accumulator Shift Left; Clear LSB; Shift MSB into C-Bit
ASR	Arithmetic Shift Right
BCC	Branch if Carry Clear
BCS	Branch if Carry Set
BEO	Branch if Equal To Zero
BFE	Branch if Greater Than or Equal To Zero
BGT	Branch if Greater Than
BHI	Branch if Higher Than
*BHS	Branch if Higher Than or Same As
BIT	Bit Test
BLE	Branch if Less Than or Equal To
*BLO	Branch if Lower Than
BLS	Branch if Lower Than or Same As
BLT	Branch if Less Than Zero
BMI	Branch if Minus
BNE	Branch if Not Equal To Zero
BPL	Branch if Plus
BRA	Branch Always
*BRN	Branch Never
BSR	Branch to Subroutine
BVC	Branch if Overflow Clear
BVS	Branch if Overflow Set
CBA	Compare Accumulators
CLC	Clear Carry
CLI	Clear Interrupt Mask
CLR	Clear
CLV	Clear Overflow
CMP	Compare
COM	Complement
CPX	Compare Index Register
*CPX	Compare Index Register; Permits Use With Any Conditional Branch Instruction
DAA	Decimal Adjust
DEC	Decrement
DES	Decrement Stack Pointer
DEX	Decrement Index Register
EOR	Exclusive OR

*F6801/F6803 Only

Table 5-3 Basic and Expanded Instruction Sets (Cont.)

Instruction	Description
INC	Increment
INS	Increment Stack Pointer
INX	Increment Index Register
JMP	Jump
JSR	Jump to Subroutine
*JSR	Additional Addressing Mode Direct
LDA	Load Accumulator
*LDD	Load Double Accumulator from Memory
LDS	Load Stack Pointer
LDX	Load Index Register
*LSL	Memory or Accumulator Shift Left; Clear LSB; Shift MSB into C-Bit
*LSLD	Double Accumulator Shift Left; Clear LSB; Shift MSB into C-Bit
LSR	Logical Shift Right
*LSRD	Double Accumulator Shift Right; Clear MSB; Shift LSB into C-Bit
*MUL	Multiply Accumulators; Leave Product in Double Accumulator
NEG	Negate
NOP	No Operation
ORA	Inclusive OR Accumulator
PSH	Push Data
*PSHX	Push Index Register to Stack
PUL	Pull Data
*PULX	Pull Index Register from Stack
RJOL	Rotate Left
ROR	Rotate Right
RTI	Return from Interrupt
RTS	Return from Subroutine
SBA	Subtract Accumulators
SBC	Subtract With Carry
SEC	Set Carry
SEI	Set Interrupt Mask
SEV	Set Overflow
STA	Store Accumulator
*STD	Store Double Accumulator
STS	Store Stack Register
STX	Store Index Register
SUB	Subtract
*SUBD	Subtract Double Accumulator; Leave Difference in Double Accumulator
SWI	Software Interrupt

*F6801/F6803 Only

Table 5-3 Basic and Expanded Instruction Sets (Cont.)

Instruction	Description
TAB	Transfer Accumulators
TAP	Transfer Accumulators to Condition Code Register
TBA	Transfer Accumulators
TPA	Transfer Condition Code Register to Accumulator
TST	Test
TSX	Transfer Stack Pointer to Index Register
TXS	Transfer Index Register to Stack Pointer
WAI	Wait for Interrupt

*F6801/F6805 Only

Table 5-4 F6809 Instruction Set

Instruction	Description
ABX	Add Accumulator B to Index Register
ADCA, ADCB	Add Memory to Accumulator With Carry
ADDA, ADDB	Add Memory to Accumulator
ADDD	Add Memory to Accumulator D
ANDA, ANDB	AND Memory With Accumulator
ANDCC	AND Condition Code Register
ASL, ASLA ASLB	Arithmetic Shift Left of Accumulator or Memory
ASR, ASRA, ASRB	Arithmetic Shift Right of Accumulator or Memory
BCC, LBCC	Branch if Carry Clear
BCS, LBCS	Branch if Carry Set
BEQ, LBEQ	Branch if Equal To
BGE, LBGE	Branch if Greater Than or Equal To
BGT, LBGT	Branch if Greater Than
BHI, LBHI	Branch if Higher
BHS, LBHS	Branch if Higher Than or Same As
BITA, BITB	Bit Test Memory With Accumulator
BLE, LBLE	Branch if Less Than or Equal To
BLO, LBLO	Branch if Lower
BLS, LBLS	Branch if Lower Than or Same As
BLT, LBLT	Branch if Less Than
BMI, LBMI	Branch if Minus
BNE, LBNE	Branch if Not Equal To
BPL, LBPL	Branch if Plus
BRA, LBRA	Branch Always
BRN, LBRN	Branch Never
BSR, LBSR	Branch to Subroutine
BVC, LBVC	Branch if Overflow Clear
BVS, LBVS	Branch if Overflow Set
CLR, CLRA, CLRB	Clear Accumulator or Memory Location
CMPA CMPB	Compare Memory from Accumulator

Table 5-4 F6809 Instruction Set (Cont.)

Instruction	Description
CMPD	Compare Memory from Accumulator D
CMPS, CMPU	Compare Memory from Stack Pointer
CMPX, CMPY	Compare Memory from Index Register
COM, COMA, COMB	Complement Accumulator or Memory Location
CWAI	AND Condition Code Register; Wait for Interrupt
DAA	Decimal Adjust Accumulator A
DEC, DECA, DECB	Decrement Accumulator or Memory Location
EORA, EORB	Exclusive OR Memory With Accumulator
EXG D, R	Exchange D With X, Y, S, U, or PC
EXG R1, R2	Exchange R1 With R2 (R1, R2 = A,B, CC, DP)
INC, INCA, INCB	Increment Accumulator or Memory Location
JMP	Jump
JSR	Jump to Subroutine
LDA, LDB	Load Accumulator from Memory
LDD	Load Accumulator D from Memory
LDS, LDU	Load Stack Pointer from Memory
LDX, LDY	Load Index Register from Memory
LEAS, LEAU	Load Effective Address into Stack Pointer
LEAX, LEAY	Load Effective Address into Index Register
LSL, LSLA, LSLB	Logical Shift Left Accumulator or Memory Location
LSR, LSRA, LSRB	Logical Shift Right Accumulator or Memory Location
MUL	Unsigned Multiply
NEG, NEGA, NEGB	Negate Accumulator or Memory
NOP	No Operation
ORA, ORB	OR Memory With Accumulator
ORCC	OR Condition Code Register
PSHS	Push A, B, CC, DP, D, X, Y, U, or PC onto hardware stack
PSHU	Push a, B, CC, DP, D, X, Y, U, or PC onto user stack
PULS	Pull A, B, CC, DP, D, X, Y, U, or PC fro hardware stack
PULU	*pull A, B, CC, DP, D, X, Y, U, or PC from user stack
ROL, ROLA, ROLB	Rotate Accumulator or Memory Left
ROR, RORA, RORB	Rotate Accumulator or Memory Right
RTI	Return from Interrupt
RTS	Return from Subroutine

Table 5-4 Instruction Set (Cont.)

Instruction	Description
SBCA, SBCB	Subtract Memory from Accumulator With Borrow
SEX	Sign Extend Accumulator B into Accumulator A
STA, STB	Store Accumulator to Memory
STD	Store Accumulator D to Memory
STS, STU	Store Stack Pointer to Memory
STX, STY	Store Index Register to Memory
SUBA, SUBB	Subtract Memory from Accumulator
SUBD	Subtract Memory from Accumulator D
SWI SWI2, SWI3	Software Interrupt
SYNC	Synchronize With Interrupt Line
TFR D, R	Transfer D to X, Y, S, U, or PC
TFR R, D	Transfer X, Y, S, U, or PC to D
TFR R1, R2	Transfer R1 to R2
TST, TSTA, TSTB	Test Accumulator or Memory Location

Descriptions

Following is data that describes the members of the F6800 microprocessor family.

FAIRCHILD
A Schlumberger Company

F6800/F68A00/F68B00

8-Bit Microprocessing Unit

Microprocessor Product

Description

The F6800 is a monolithic 8-bit microprocessing unit (MPU) forming the central control function for the Fairchild F6800 family. Compatible with TTL, the F6800, as with all F6800 system parts, requires only one +5.0 V power supply and no external TTL devices for bus interface.

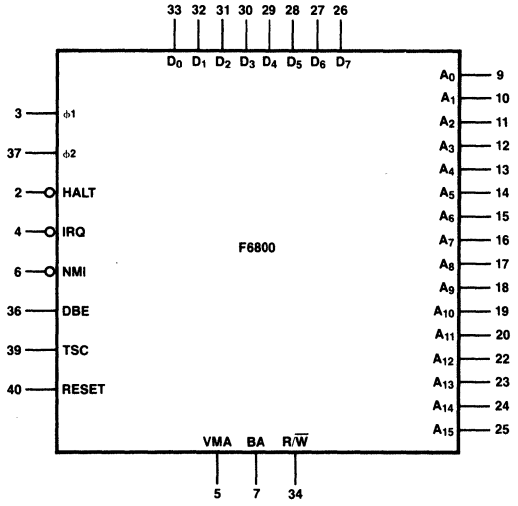
The F6800 is capable of addressing 65K bytes of memory with its 16-bit address lines. The 8-bit data bus is bidirectional as well as 3-state, making direct memory addressing and multiprocessing applications realizable.

- **8-Bit Parallel Processing**
- **Bidirectional Data Bus**
- **16-Bit Address Bus — 65K Bytes of Addressing**
- **72 Instructions — Variable Length**
- **7 Addressing Modes — Direct, Relative, Immediate, Indexed, Extended, Implied and Accumulator**
- **Variable Length Stack**
- **Vectored Restart**
- **Maskable Interrupt Vector**
- **Separate Non-Maskable Interrupt — Internal Registers Saved in Stack**
- **6 Internal Registers — 2 Accumulators, Index Register, Program Counter, Stack Pointer, and Condition Code Register**
- **Direct Memory Addressing (DMA) and Multiple Processor Capability**
- **Simplified Clocking Characteristics**
- **Clock Rates 1 MHz (F6800), 1.5 MHz (F68A00), and 2 MHz (F68B00)**
- **Simple Bus Interface Without TTL**
- **Halt and Single Instruction Execution Capability**

Pin Names

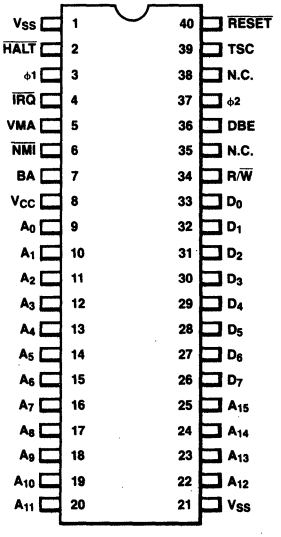
$\overline{D_0}$-$\overline{D_7}$	Bidirectional Data Bus
$\overline{HALT}$	Halt Input
$\phi 1$, $\phi 2$	Clock Inputs
$\overline{IRQ}$	Interrupt Request Input
$\overline{NMI}$	Non-Maskable Interrupt Input
DBE	Data Bus Enable Input
TSC	3-State Control Input
$\overline{RESET}$	Reset Input
VMA	Valid Memory Address Output
BA	Bus Available Output
A_0-A_{15}	Address Bus Outputs
R/$\overline{W}$	Read/Write Output
V_{CC}	+5 V Power Supply Input
V_{SS}	Ground

Logic Symbol



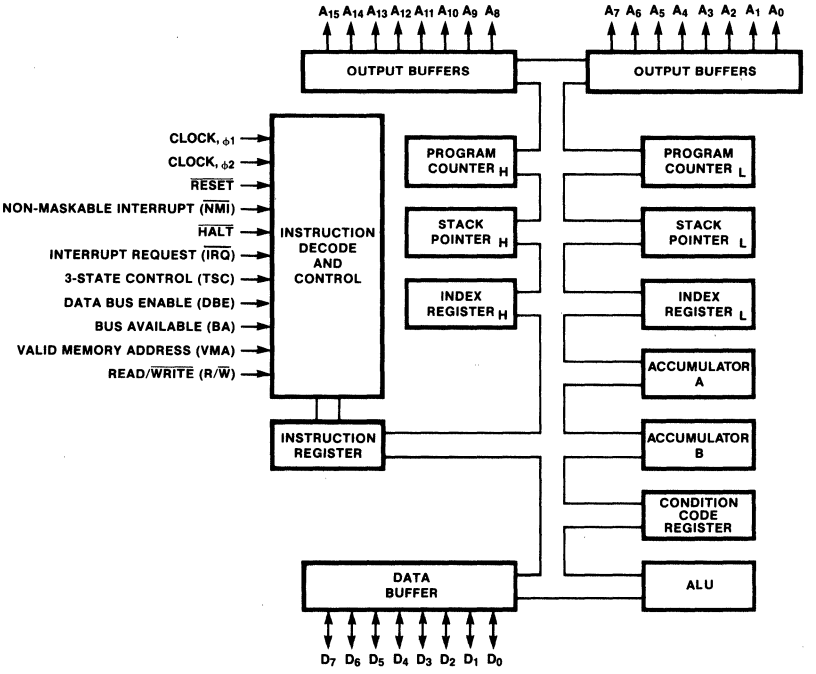
V_{CC} = Pin 8
V_{SS} = Pins 1, 21

Connection Diagram
40-Pin DIP



(Top View)

Block Diagram



MPU Signal Description

Proper operation of the MPU requires that certain control and timing signals be provided to accomplish specific functions and that other signal lines be monitored to determine the state of the processor.

Clocks Phase One and Phase Two (ϕ1,ϕ2)
Two pins are used for a 2-phase non-overlapping clock that runs at the V_{CC} voltage level.

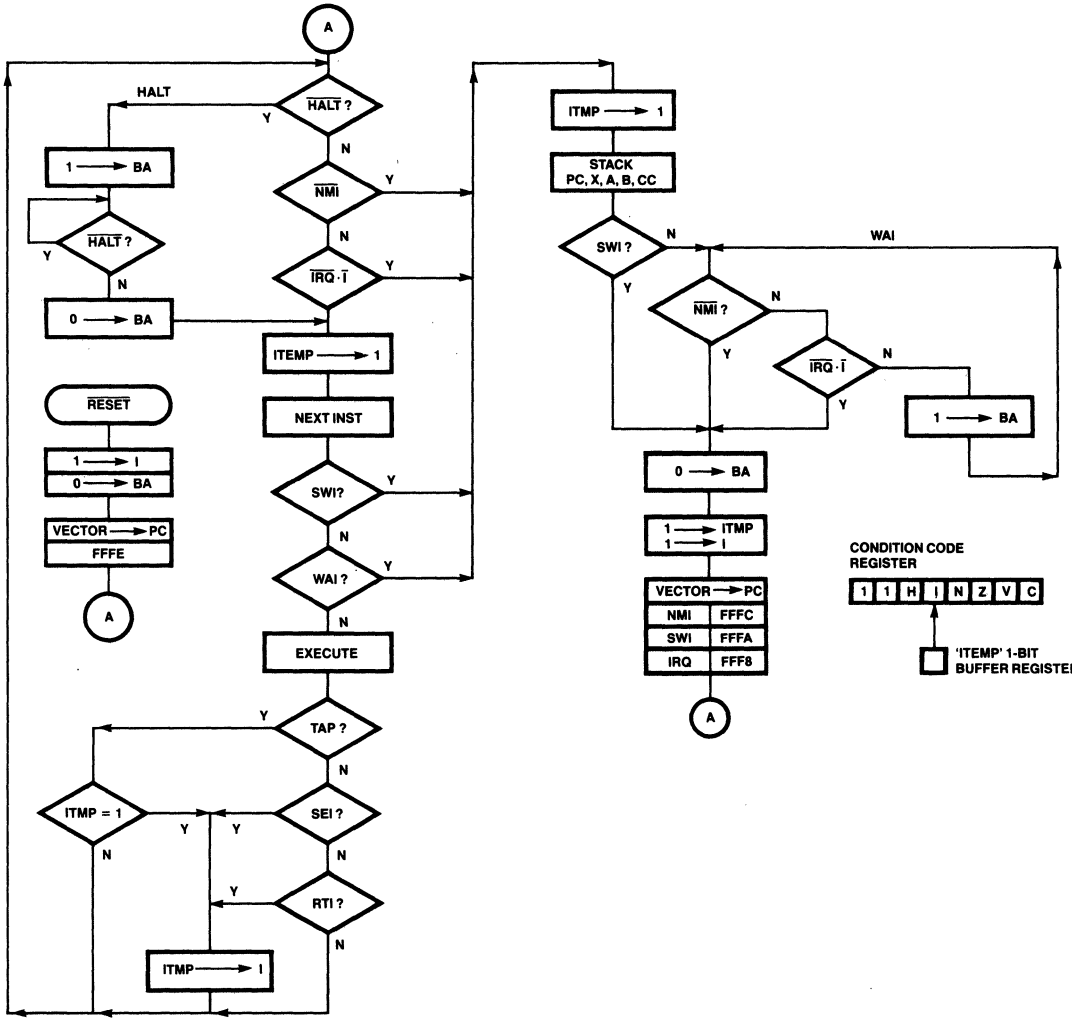
Figure 27 shows the microprocessor clocks, and the *Clock Timing* table shows the static and dynamic clock specifications. The HIGH level is specified at V_{IHC} and the LOW level is specified at V_{ILC}. The allowable clock frequency is specified by f (frequency). The minimum ϕ1 and ϕ2 HIGH level pulse widths are specified by $PW_{\phi H}$ (pulse width HIGH time). To guarantee the required access time for the peripherals, the clock up time, t_{ut}, is specified. Clock separation, t_d, is measured at a maximum voltage of V_{OV} (overlap voltage). This allows for a multitude of clock variations at the system frequency rate.

Address Bus (A_0–A_{15})
Sixteen pins are used for the address bus. The outputs are 3-state bus drivers capable of driving one standard TTL load and 90 pF. When the output is turned off, it is essentially an open circuit. This permits the MPU to be used in DMA applications. Putting TSC in its HIGH state forces the address bus to go into the 3-state mode.

Data Bus (D_0–D_7)
Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has 3-state output buffers capable of driving one standard TTL load and 130 pF. The data bus is placed in the 3-state mode when DBE is LOW.

Data Bus Enable (DBE)
This input is the 3-state control signal for the MPU data bus and will enable the bus drivers when in the HIGH state. This input is TTL-compatible; however, in normal operation it would be driven by the phase two clock. During an MPU read cycle, the data bus drivers will be disabled internally. When it is desired that another device control the data bus, such as

Fig. 1 MPU Flow Chart



Notes

1. Reset is recognized at any position in the flowchart.
2. Instructions which affect the I-Bit act upon a one-bit buffer register, "ITMP". This has the effect of delaying any clearing of the I-Bit one clock time. Setting the I-Bit, however, is not delayed.
3. Refer to tables 8 through 13 for details of instruction execution.

in Direct Memory Access (DMA) applications, DBE should be held LOW.

If additional data set-up or hold time is required on an MPU write, the DBE down time can be decreased as shown in *Figure 29* (DBE ≠ ϕ2). The minimum down time for DBE is $t_{\overline{DBE}}$ as shown and must occur within ϕ1 up time. The minimum delay from the trailing edge of DBE to the trailing edge of ϕ1 is $t_{\overline{DBED}}$. By skewing DBE with respect to E in this manner, data set-up or hold time can be increased.

Bus Available (BA)
The Bus Available signal will normally be in the LOW state; when activated, it will go to the HIGH state, indicating that the microprocessor has stopped and that the address bus is available. This will occur if the $\overline{HALT}$ line is in the LOW state or the processor is in the WAIT state as a result of the execution of a WAIT instruction. At such time, all 3-state output drivers will go to their OFF state and other outputs to their normally inactive level. The processor is removed from the WAIT state by the occurrence of a maskable (mask bit I = "0") or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30 pF. If TSC is in the HIGH state, Bus Available will be LOW.

Read/Write (R/$\overline{W}$)
This TTL-compatible output signals the peripherals and memory devices whether the MPU is in a Read (HIGH) or Write (LOW) state. The normal standby state of this signal is Read (HIGH). 3-State Control (TSC) going HIGH will turn Read/$\overline{Write}$ to the OFF (high-impedance) state. Also, when the processor is halted, it will be in the OFF state. This output is capable of driving one standard TTL load and 90 pF.

Reset ($\overline{RESET}$)
The $\overline{RESET}$ input is used to reset and start the MPU from a power-down condition resulting from a power failure or initial start-up of the processor. This input can also be used to reinitialize the machine at any time after start-up.

If a HIGH level is detected in this input, this will signal the MPU to begin the reset sequence. During the reset sequence, the contents of the last two locations (FFFE, FFFF) in memory will be loaded into the program counter to point to the beginning of the reset routine. During the reset routine, the interrupt mask bit is set and must be cleared under program control before the MPU can be interrupted by $\overline{IRQ}$. While $\overline{RESET}$ is LOW (assuming a minimum of eight clock cycles have occurred) the MPU output signals will be in the following states: VMA = LOW, BA = LOW, data bus = high impedance, R/$\overline{W}$ = HIGH (read state), and the address bus will contain the reset address FFFE. *Figure 2* illustrates a power-up sequence using the $\overline{RESET}$ control line. After the power supply reaches 4.75 V a minimum of eight clock cycles are required for the processor to stabilize in preparation for restarting. During these eight cycles, VMA will be in an indeterminate state so any devices that are enabled by VMA which could accept a false write during this time (such as a battery-backed RAM) must be disabled until VMA is forced LOW after eight cycles. $\overline{RESET}$ can go HIGH asynchronously with the system clock any time after the eighth cycle.

Reset timing is shown in *Figure 2* and the *Read/Write Timing* table. The maximum rise and fall transition times are specified by t_{PCr} and t_{PCf}. If $\overline{RESET}$ is HIGH at t_{PCS} (processor control set-up time) as shown in *Figure 2* in any given cycle, then the restart sequence will begin on the next cycle as shown. The $\overline{RESET}$ control line may also be used to reinitialize the MPU system at any time during its operation. This is accomplished by pulsing $\overline{RESET}$ LOW for the duration of a minimum of three complete ϕ2 cycles. The Reset pulse can be completely asynchronous with the MPU system clock and will be recognized during ϕ2 if set-up time t_{PCS} is met.

Interrupt Request ($\overline{IRQ}$)
This level-sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that is being executed before it recognizes the request. At that time, if the interrupt mask bit in the condition code register is not set, the machine will begin an interrupt sequence. The index register, program counter, accumulators, and condition code register are stored away on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit HIGH so that further interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFF8 and FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory. Interrupt timing is shown in *Figure 3*.

The $\overline{HALT}$ line must be in the HIGH state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is LOW.

The $\overline{IRQ}$ has a high-impedance pullup device internal to the chip; however, a 3 kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

Non-Maskable Interrupt ($\overline{NMI}$) and Wait for Interrupt (WAI)
The F6800 is capable of handling two types of interrupts: maskable ($\overline{IRQ}$) as described earlier, and non-maskable ($\overline{NMI}$). $\overline{IRQ}$ is maskable by the interrupt mask in the condition code register while $\overline{NMI}$ is not maskable. The handling of these interrupts by the MPU is the same except that each has its own vector address. The behavior of the MPU when interrupted is shown in *Figure 3* which details the MPU response to an interrupt while the MPU is executing the

Fig. 2 Reset Timing

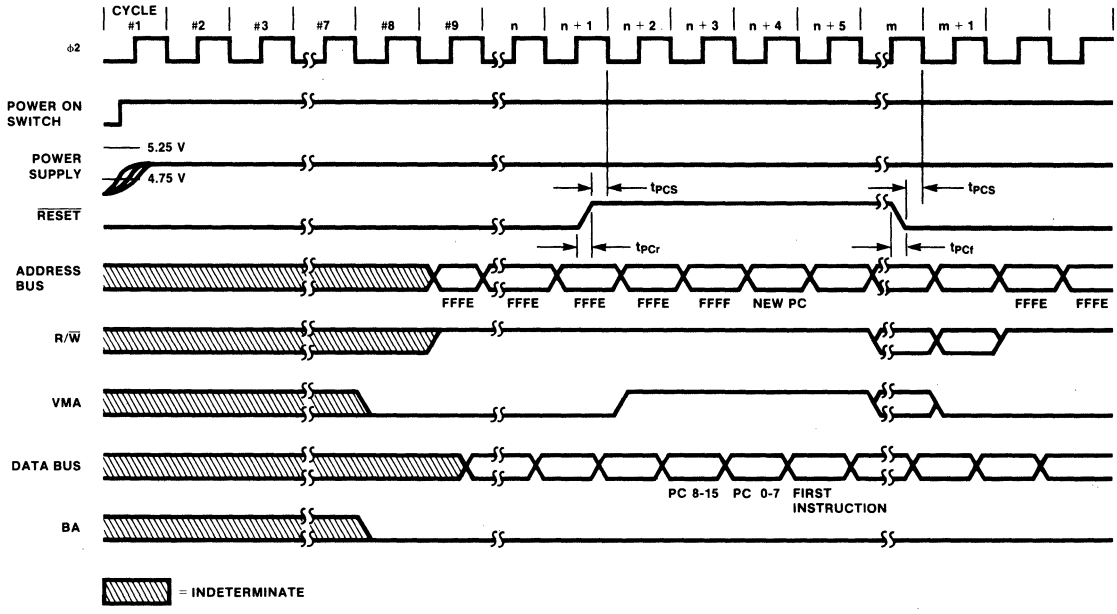
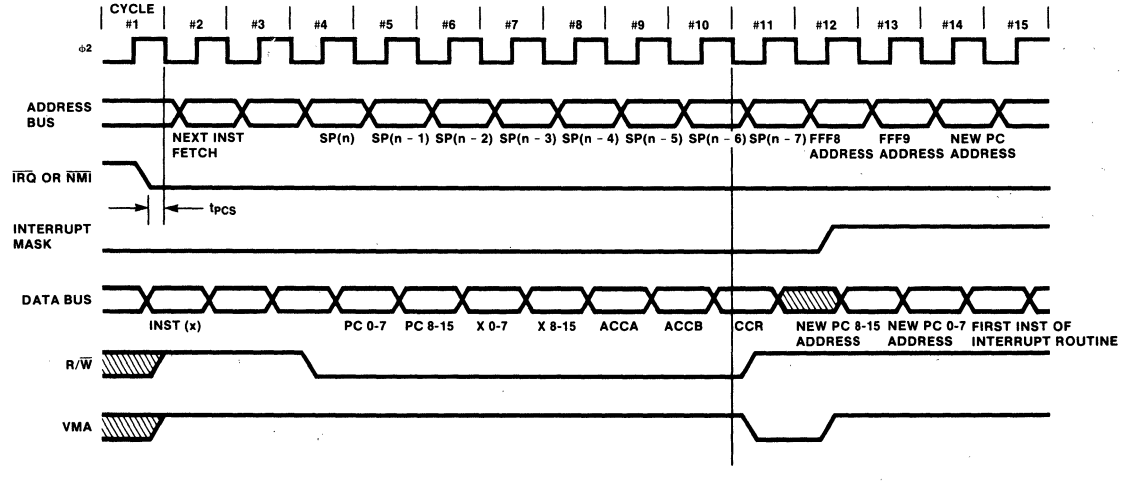


Fig. 3 Interrupt Timing



control program. The interrupt shown could be either $\overline{IRQ}$ or $\overline{NMI}$ and can be asynchronous with respect to $\phi 2$. The interrupt is shown going LOW at time t_{PCS} in cycle #1 which precedes the first cycle of an instruction (OP code fetch). This instruction is not executed, but instead, the program counter (PC), index register (IX), accumulators (ACCX), and the condition code register (CCR) are pushed onto the stack.

The Interrupt Mask bit is set to prevent further interrupts. The address of the interrupt service routine is then fetched from FFFC, FFFD for an $\overline{NMI}$ interrupt and from FFF8, FFF9 for an $\overline{IRQ}$ interrupt. Upon completion of the interrupt service routine, the execution of RTI will pull the PC, IX, ACCX, and CCR off of the stack; the Interrupt Mask bit is restored to its condition prior to Interrupts.

Figure 4 is a similar interrupt sequence, except in this case, a WAIT instruction has been executed in preparation for the interrupt. This technique speeds up the MPU's response to the interrupt because the stacking of the PC, IX, ACCX, and the CCR is already done. While the MPU is waiting for the interrupt, Bus Available will go HIGH indicating the following states of the control lines: VMA is LOW, and the address bus, R/$\overline{W}$ and data bus are all in the high impedance state. After the interrupt occurs, it is serviced as previously described.

Table 1 Memory Map for Interrupt Vectors

Vector		
MS	**LS**	**Description**
FFFE	FFFF	Restart
FFFC	FFFD	Non-maskable Interrupt
FFFA	FFFB	Software Interrupt
FFF8	FFF9	Interrupt Request

Refer to *Figure 4* for program flow for Interrupts.

3-State Control (TSC)
When the 3-State Control (TSC) line is a logic "1", the address bus and the R/$\overline{W}$ line are placed in a high impedance state. VMA and BA are forced LOW when TSC = "1" to prevent false reads or writes on any device enabled by VMA. It is necessary to delay program execution while TSC is held HIGH. This is done by insuring that no transitions of $\phi 1$ (or $\phi 2$) occur during this period. (Logic levels of the clocks are irrelevant so long as they do not change.) Since the MPU is a dynamic device, the $\phi 1$ clock can be stopped for a maximum time $PW_{\phi H}$ without destroying data within the MPU. TSC then can be used in a short Direct Memory Access (DMA) application.

Figure 5 shows the effect of TSC on the MPU. TSC must have its transitions at t_{TSE} (3-state enable) while holding $\phi 1$ HIGH and $\phi 2$ LOW as shown. The address bus and R/$\overline{W}$ line will reach the high impedance state at t_{TSD} (3-state delay), with VMA being forced LOW. In this example, the data bus is also in the high impedance state while $\phi 2$ is being held LOW since DBE = $\phi 2$. At this time, a DMA transfer could occur on cycles #3 and #4. When TSC is returned LOW, the MPU address and R/$\overline{W}$ lines return to the bus. Because it is too late in cycle #5 to access memory, this cycle is dead and used for synchronization. Program execution resumes in cycle #6.

Valid Memory Address (VMA)
This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not 3-state. One standard TTL load and 90 pF may be directly driven by this active HIGH signal.

$\overline{HALT}$
When this level sensitive input is in the LOW state, all activity in the machine will be halted.

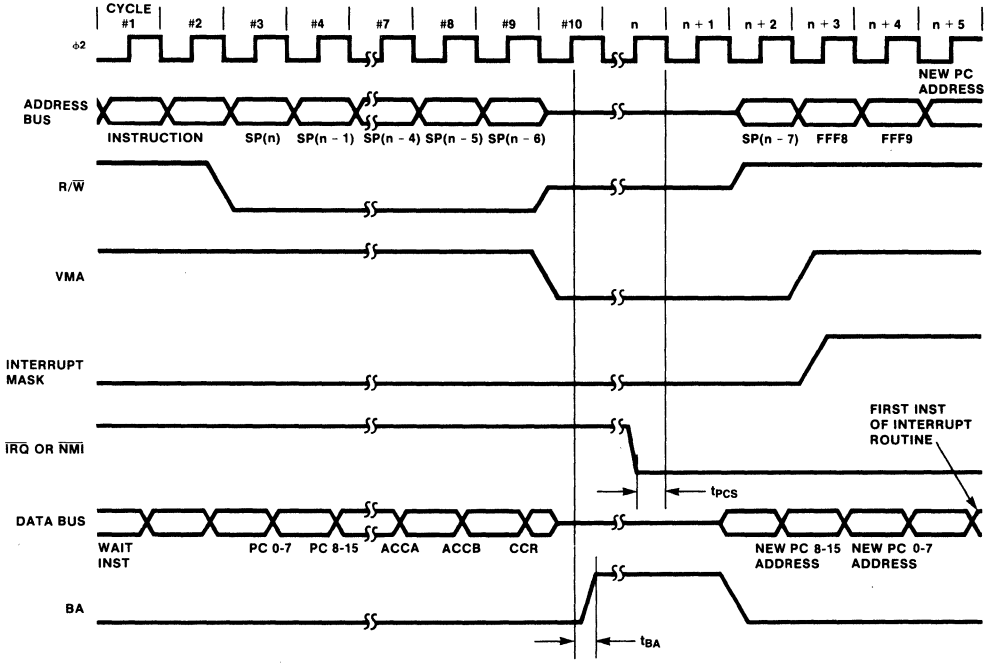
The $\overline{HALT}$ line provides an input to the MPU to allow control of program execution by an outside source. If $\overline{HALT}$ is HIGH, the MPU will execute the instructions; if it is LOW, the MPU will go to a halted, or idle, mode. A response signal, Bus Available (BA) provides an indication of the current MPU status. When BA is LOW, the MPU is in the process of executing the control program; if BA is HIGH, the MPU has halted and all internal activity has stopped.

When BA is HIGH, the address bus, data bus, and R/$\overline{W}$ line will be in a high impedance state, effectively removing the MPU from the system bus. VMA is forced LOW so that the floating system bus will not activate any device on the bus that is enabled by VMA.

While the MPU is halted, all program activity is stopped, and if either an $\overline{NMI}$ or $\overline{IRQ}$ interrupt occurs, it will be latched into the MPU and acted on as soon as the MPU is taken out of the halted mode. If a $\overline{RESET}$ command occurs while the MPU is halted, the following states occur: VMA = LOW, BA = LOW, data bus = high impedance, R/$\overline{W}$ = HIGH (read state), and the address bus will contain address FFFE as long as $\overline{RESET}$ is LOW. As soon as the $\overline{HALT}$ line goes HIGH, the MPU will go to locations FFFE and FFFF for the address of the reset routine.

Figure 6 shows the timing relationships involved when halting the MPU. The instruction illustrated is a 1-byte, 2-cycle instruction such as CLRA. When $\overline{HALT}$ goes LOW, the MPU will halt after completing execution of the current instruction. The transition of $\overline{HALT}$ must occur t_{PCS} before the trailing edge of $\phi 1$ of the last cycle of an instruction (Point A of

Fig. 4 Wait Instruction Timing



Note
Midrange waveform indicates high-impedance state.

Fig. 5 3-State Control Timing

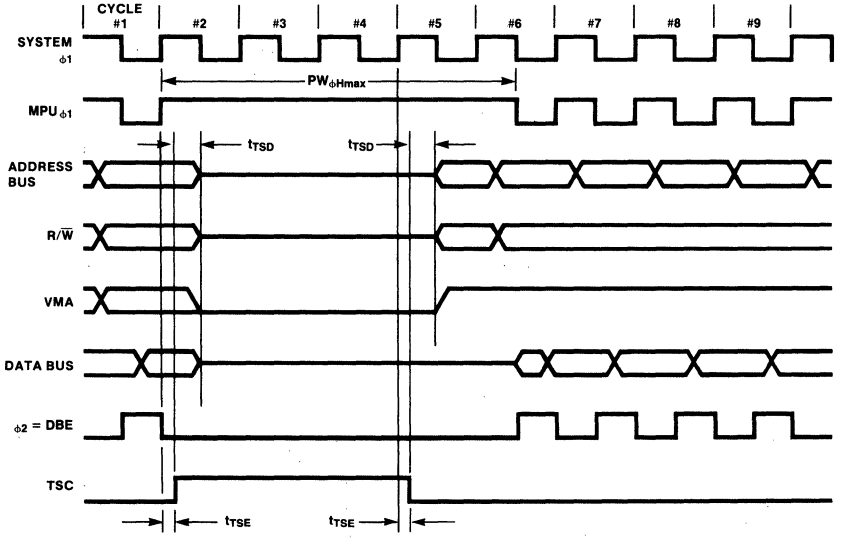
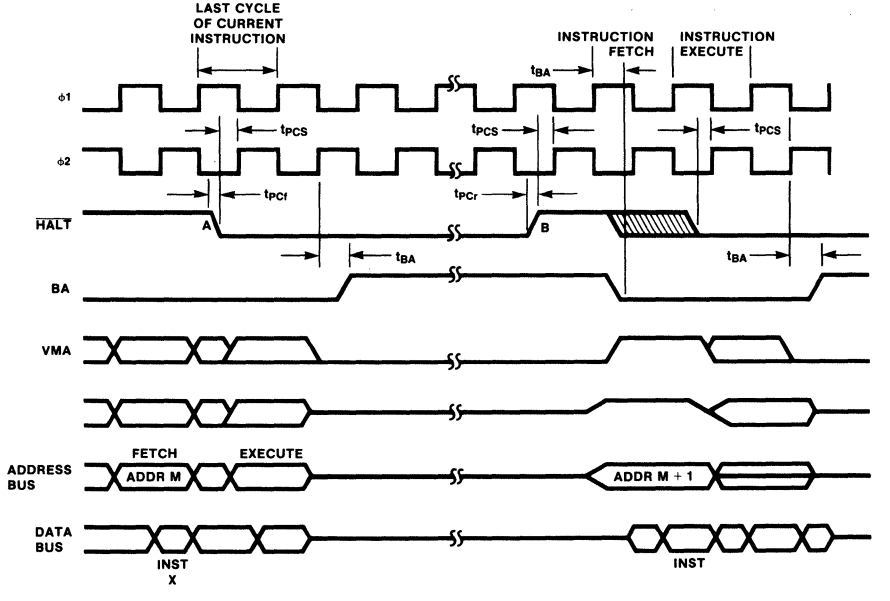


Fig. 6 Halt and Single Instruction Execution for System Debug



Note
Midrange waveform indicates high impedance state.

Figure 6). $\overline{HALT}$ must not go LOW any time later than the minimum t_{PCS} specified.

The fetch of the OP code by the MPU is the first cycle of the instruction. If $\overline{HALT}$ had not been LOW at Point A, but went LOW during $\phi 2$ of that cycle, the MPU would have halted after completion of the following instruction. BA will go HIGH by time t_{BA} (bus available delay time) after the last instruction cycle. At this time, VMA is LOW and R/$\overline{W}$, address bus, and the data bus are in the high-impedance state.

To debug programs it is advantageous to step through programs instruction by instruction. To do this, $\overline{HALT}$ must be brought HIGH for one MPU cycle and then returned LOW as shown at Point B of *Figure 6*. Again, the transitions of $\overline{HALT}$ must occur t_{PCS} before the trailing edge of the next $\phi 1$, indicating that the Address Bus, Data Bus, VMA and R/$\overline{W}$ lines are back on the bus. A single-byte, 2-cycle instruction such as LSR is used for this example also. During the first cycle, the instruction Y is fetched from address M + 1. BA returns HIGH at t_{BA} on the last cycle of the instruction indicating the MPU is off the bus. If instruction Y had been three cycles, the width of the BA LOW time would have been increased by one cycle.

MPU Registers

The MPU has three 16-bit registers and three 8-bit registers available for use by the programmer (*Figure 7*).

Program Counter
The program counter is a 2-byte (16 bits) register that points to the current program address.

Stack Pointer
The stack pointer is a 2-byte register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access read/write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be nonvolatile.

Index Register
The index register is a 2-byte register that is used to store data or a 16-bit memory address for the Indexed mode of memory addressing.

Accumulators
The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit (ALU).

Condition Code Register
The condition code register indicates the results of an arithmetic logic unit operation: negative (N), zero (Z), overflow (V), carry from bit 7 (C), and half carry from bit 3 (H). These bits of the condition code register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The unused bits of the condition code register (bit 6 and bit 7) are ones.

MPU Instruction Set

The F6800 instructions are described in detail in the F6800 Programming Manual. This section will provide a brief introduction and discuss their use in developing F6800 control programs. The F6800 has a set of 72 different executable source instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions.

5

Each of the 72 executable instructions of the source language assembles into one to three bytes of machine code. The number of bytes depends on the particular instruction and on the addressing mode. (The addressing modes which are available for use with the various executive instructions are discussed later.)

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 72 instructions in all valid modes of addressing, are shown in *Table 2*. There are 197 valid machine codes, 59 of the 256 possible codes being unassigned.

When an instruction translates into two or three bytes of code, the second byte, or the second and third bytes contain(s) an operand, an address, or information from which an address is obtained during execution.

Microprocessor instructions are often divided into three general classifications: (1) memory reference, so called because they operate on specific memory locations; (2) operating instructions that function without needing a memory reference; (3) I/O instructions for transferring data between the microprocessor and peripheral devices.

In many instances, the F6800 performs the same operation on both its internal accumulators and the external memory locations. In addition, the F6800 interface adapters (PIA and ACIA) allow the MPU to treat peripheral devices exactly like other memory locations, hence, no I/O instructions as such are required. Because of these features, other classifications are more suitable for introducing the F6800's instruction set: (1) accumulator and memory operations; (2) program control operations; (3) condition code register operations.

Fig. 7 Programming Model of The Microprocessing Unit

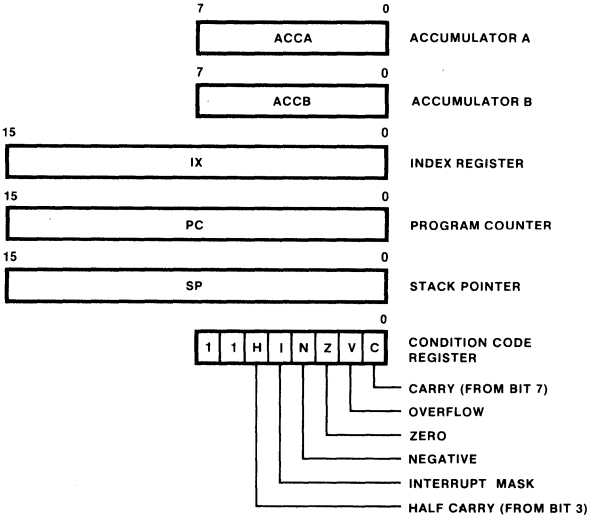


Table 2 Microprocessor Instruction Set—Alphabetic Sequence

ABA	Add Accumulators
ADC	Add with Carry
ADD	Add
AND	Logical And
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
BCC	Branch if Carry Clear
BCS	Branch if Carry Set
BEQ	Branch if Equal to Zero
BGE	Branch if Greater or Equal Zero
BGT	Branch if Greater than Zero
BHI	Branch if Higher
BIT	Bit Test
BLE	Branch if Less or Equal
BLS	Branch if Lower or Same
BLT	Branch if Less than Zero
BMI	Branch if Minus
BNE	Branch if Not Equal to Zero
BPL	Branch if Plus
BRA	Branch Always
BSR	Branch to Subroutine
BVC	Branch if Overflow Clear
BVS	Branch if Overflow Set
CBA	Compare Accumulators
CLC	Clear Carry
CLI	Clear Interrupt Mask
CLR	Clear
CLV	Clear Overflow
CMP	Compare
COM	Complement
CPX	Compare Index Register
DAA	Decimal Adjust
DEC	Decrement
DES	Decrement Stack Pointer
DEX	Decrement Index Register
EOR	Exclusive OR
INC	Increment
INS	Increment Stack Pointer
INX	Increment Index Register
JMP	Jump
JSR	Jump to Subroutine
LDA	Load Accumulator
LDS	Load Stack Pointer
LDX	Load Index Register
LSR	Logical Shift Right
NEG	Negate
NOP	No Operation
ORA	Inclusive OR Accumulator
PSH	Push Data
PUL	Pull Data
ROL	Rotate Left
ROR	Rotate Right
RTI	Return from Interrupt
RTS	Return from Subroutine
SBA	Subtract Accumulators
SBC	Subtract with Carry
SEC	Set Carry
SEI	Set Interrupt Mask
SEV	Set Overflow
STA	Store Accumulator
STS	Store Stack Register
STX	Store Index Register
SUB	Subtract
SWI	Software Interrupt
TAB	Transfer Accumulators
TAP	Transfer Accumulators to Condition Code Reg.
TBA	Transfer Accumulators
TPA	Transfer Condition Code Reg. to Accumulator
TST	Test
TSX	Transfer Stack Pointer to Index Register
TXS	Transfer Index Register to Stack Pointer
WAI	Wait for Interrupt

Table 3 Hexadecimal Values of Machine Codes

00	*		3B	RTI			76	ROR		EXT	B1	CMP	A	EXT	EC	*							
01	NOP		3C	*			77	ASR		EXT	B2	SBC	A	EXT	ED	*							
02	*		3D	*			78	ASL		EXT	B3	*			EE	LDX		IND					
03	*		3E	WAI			79	ROL		EXT	B4	AND	A	EXT	EF	STX		IND					
04	*		3F	SWI			7A	DEC		EXT	B5	BIT	A	EXT	F0	SUB	B	EXT					
05	*		40	NEG	A		7B	*			B6	LDA	A	EXT	F1	CMP	B	EXT					
06	TAP		41	*			7C	INC		EXT	B7	STA	A	EXT	F2	SBC	B	EXT					
07	TPA		42	*			7D	TST		EXT	B8	EOR	A	EXT	F3	*							
08	INX		43	COM	A		7E	JMP		EXT	B9	ADC	A	EXT	F4	AND	B	EXT					
09	DEX		44	LSR	A		7F	CLR		EXT	BA	ORA	A	EXT	F5	BIT	B	EXT					
0A	CLV		45	*			80	SUB	A	IMM	BB	ADD	A	EXT	F6	LDA	B	EXT					
0B	SEV		46	ROR	A		81	CMP	A	IMM	BC	CPX		EXT	F7	STA	B	EXT					
0C	CLC		47	ASR	A		82	SBC	A	IMM	BD	JSR		EXT	F8	ADC	B	EXT					
0D	SEC		48	ASL	A		83	*			BE	LDS		EXT	F9	ADC	B	EXT					
0E	CLI		49	ROL	A		84	AND	A	IMM	BF	STS		EXT	FA	ORA	B	EXT					
0F	SEI		4A	DEC	A		85	BIT	A	IMM	C0	SUB	B	IMM	FB	ADD	B	EXT					
10	SBA		4B	*			86	LDA	A	IMM	C1	CMP	B	IMM	FC	*							
11	CBA		4C	INC	A		87	*			C2	SBC	B	IMM	FD	*							
12	*		4D	TST	A		88	EOR	A	IMM	C3	*			FE	LDX		EXT					
13	*		4E	*			89	ADC	A	IMM	C4	AND	B	IMM	FF	STX		EXT					
14	*		4F	CLR	A		8A	ORA	A	IMM	C5	BIT	B	IMM									
15	*		50	NEG	B		8B	ADD	A	IMM	C6	LDA	B	IMM									
16	TAB		51	*			8C	CPX	A	IMM	C7	*											
17	TBA		52	*			8D	BSR		REL	C8	EOR	B	IMM									
18	*		53	COM	B		8E	LDS		IMM	C9	ADC	B	IMM									
19	DAA		54	LSR	B		8F	*			CA	ORA	B	IMM									
1A	*		55	*			90	SUB	A	DIR	CB	ADD	B	IMM									
1B	ABA		56	ROR	B		91	CMP	A	DIR	CC	*											
1C	*		57	ASR	B		92	SBC	A	DIR	CD	*											
1D	*		58	ASL	B		93	*			CE	LDX		IMM									
1E	*		59	ROL	B		94	AND	A	DIR	CF	*											
1F	*		5A	DEC	B		95	BIT	A	DIR	D0	SUB	B	DIR									
20	BRA	REL	5B	*			96	LDA	A	DIR	D1	CMP	B	DIR									
21	*		5C	INC	B		97	STA	A	DIR	D2	SBC	B	DIR									
22	BHI	REL	5D	TST	B		98	EOR	A	DIR	D3	*											
23	BLS	REL	5E	*			99	ADC	A	DIR	D4	AND	B	DIR									
24	BCC	REL	5F	CLR	B		9A	ORA	A	DIR	D5	BIT	B	DIR									
25	BCS	REL	60	NEG		IND	9B	ADD	A	DIR	D6	LDA	B	DIR									
26	BNE	REL	61	*			9C	CPX		DIR	D7	STA	B	DIR									
27	BEQ	REL	62	*			9D	*			D8	EOR	B	DIR									
28	BVC	REL	63	COM		IND	9E	LDS		DIR	D9	ADC	B	DIR									
29	BVS	REL	64	LSR		IND	9F	STS		DIR	DA	ORA	B	DIR									
2A	BPL	REL	65	*			A0	SUB	A	IND	DB	ADD	B	DIR									
2B	BMI	REL	66	ROR		IND	A1	CMP	A	IND	DC	*											
2C	BGE	REL	67	ASR		IND	A2	SBC	A	IND	DD	*											
2D	BLT	REL	68	ASL		IND	A3	*			DE	LDX		DIR									
2E	BGT	REL	69	ROL		IND	A4	AND	A	IND	DF	STX		DIR									
2F	BLE	REL	6A	DEC		IND	A5	BIT	A	IND	E0	SUB	B	IND									
30	TSX		6B	*			A6	LDA	A	IND	E1	CMP	B	IND									
31	INS		6C	INC		IND	A7	STA	A	IND	E2	SBC	B	IND									
32	PUL	A	6D	TST		IND	A8	EOR	A	IND	E3	*											
33	PUL	B	6E	JMP		IND	A9	ADC	A	IND	E4	AND	B	IND									
34	DES		6F	CLR		IND	AA	ORA	A	IND	E5	BIT	B	IND									
35	TXS		70	NEG		EXT	AB	ADD	A	IND	E6	LDA	B	IND									
36	PSH	A	71	*			AC	CPX		IND	E7	STA	B	IND									
37	PSH	B	72				AD	JSR		IND	E8	EOR	B	IND									
38	*		73	COM		EXT	AE	LDS		IND	E9	ADC	B	IND									
39	RTS		74	LSR		EXT	AF	STS		IND	EA	ORA	B	IND									
3A	*		75	*			B0	SUB	A	EXT	EB	ADD	B	IND									

Notes

1. Addressing Modes: A = Accumulator A; B = Accumulator B; IMM = Immediate; DIR = Direct; REL = Relative; IND = Indexed
2. Unassigned code indicated by an asterisk (*)

Table 4 Accumulator and Memory Operations

The accumulator and memory operations and their effect on the CCR are shown in *Table 4*. Included are Arithmetic Logic, Data Test and Data Handling instructions.

Operations	Mnemonic	Addressing Modes																Boolean/Arithmetic Operation (All register labels refer to contents)	Cond. Code Reg. *					
		Immed			Direct			Index			Extnd			Implied					5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#			H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3					A + M → A	↕	●	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3					B + M → B	↕	●	↕	↕	↕	↕
Add Acmltrs	ABA													1B	2	1		A + B → A	↕	●	↕	↕	↕	↕
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3					A + M + C → A	↕	●	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3					B + M + C → B	↕	●	↕	↕	↕	
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3					A • M → A	●	●	↕	↕	R	●
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3					B • M → B	●	●	↕	↕	R	●
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3					A • M	●	●	↕	↕	R	●
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3					B • M	●	●	↕	↕	R	●
Clear	CLR							6F	7	2	7F	6	3					00 → M	●	●	R	S	R	R
	CLRA													4F	2	1		00 → A	●	●	R	S	R	R
	CLRB													5F	2	1		00 → B	●	●	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3					A - M	●	●	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3					B - M	●	●	↕	↕	↕	↕
Compare Acmltrs	CBA													11	2	1		A - B	●	●	↕	↕	↕	↕
Complement, 1s	COM							63	7	2	73	6	3					M → M	●	●	↕	↕	R	S
	COMA													43	2	1		A → A	●	●	↕	↕	R	S
	COMB													53	2	1		B → B	●	●	↕	↕	R	S
Complement, 2s	NEG							60	7	2	70	6	3					00 - M → M	●	●	↕	↕	1	2
(Negate)	NEGA													40	2	1		00 - A → A	●	●	↕	↕	1	2
	NEGB													50	2	1		00 - B → B	●	●	↕	↕	1	2
Decimal Adjust, A	DAA													19	2	1		Converts Binary Add. of BCD Characters into BCD Format	●	●	↕	↕	↕	3
Decrement	DEC							6A	7	2	7A	6	3					M - 1 → M	●	●	↕	↕	4	●
	DECA													4A	2	1		A - 1 → A	●	●	↕	↕	4	●
	DECB													5A	2	1		B - 1 → B	●	●	↕	↕	4	●
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3					A + M → A	●	●	↕	↕	R	●
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3					B + M → B	●	●	↕	↕	R	●
Increment	INC							6C	7	2	7C	6	3					M + 1 → M	●	●	↕	↕	5	●
	INCA													4C	2	1		A + 1 → A	●	●	↕	↕	5	●
	INCB													5C	2	1		B + 1 → B	●	●	↕	↕	5	●
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3					M → A	●	●	↕	↕	R	●
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3					M → B	●	●	↕	↕	R	●
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3					A + M → A	●	●	↕	↕	R	●
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3					B + M → B	●	●	↕	↕	R	●
Push Data	PSHA													36	4	1		A → M_{SP}, SP - 1 → SP	●	●	●	●	●	●
	PSHB													37	4	1		B → M_{SP}, SP - 1 → SP	●	●	●	●	●	●
Pull Data	PULA													32	4	1		SP + 1 → SP, M_{SP} → A	●	●	●	●	●	●
	PULB													33	4	1		SP + 1 → SP, M_{SP} → B	●	●	●	●	●	●

Table 4 Accumulator and Memory Operations (Cont.)

Operations	Mnemonic	Addressing Modes: Immed OP	~	#	Direct OP	~	#	Index OP	~	#	Extnd OP	~	#	Implied OP	~	#	Boolean/Arithmetic Operation (All register labels refer to contents)	Cond. Code Reg. * 5 H	4 I	3 N	2 Z	1 V	0 C
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	↕	↕	6	↕
	ROLA													49	2	1	A	•	•	↕	↕	6	↕
	ROLB													59	2	1	B	•	•	↕	↕	6	↕
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	↕	↕	6	↕
	RORA													46	2	1	A	•	•	↕	↕	6	↕
	RORB													56	2	1	B	•	•	↕	↕	6	↕
Shift Left,	ASL							68	7	2	78	6	3				M	•	•	↕	↕	6	↕
Arithmetic	ASLA													48	2	1	A	•	•	↕	↕	6	↕
	ASLB													58	2	1	B	•	•	↕	↕	6	↕
Shift Right,	ASR							67	7	2	77	6	3				M	•	•	↕	↕	6	↕
Arithmetic	ASRA													47	2	1	A	•	•	↕	↕	6	↕
	ASRB													57	2	1	B	•	•	↕	↕	6	↕
Shift Right,	LSR							64	7	2	74	6	3				M	•	•	R	↕	6	↕
Logic	LSRA													44	2	1	A	•	•	R	↕	6	↕
	LSRB													54	2	1	B	•	•	R	↕	6	↕
Store Acmltr	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
Subtract	SUBA	60	2	2	90	3	2	A0	5	2	80	4	3				A - M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B - M → B	•	•	↕	↕	↕	↕
Subtract Acmltrs	SBA													10	2	1	A - B → A	•	•	↕	↕	↕	↕
Subtr. with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A - M - C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B - M - C → B	•	•	↕	↕	↕	↕
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero	TST							6D	7	2	7D	6	3				M - 00	•	•	↕	↕	R	R
or Minus	TSTA													4D	2	1	A - 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B - 00	•	•	↕	↕	R	R
																		H	I	N	Z	V	C

Note
Accumulator addressing mode instructions are included in the column for IMPLIED addressing
*See condition code register notes page 26

Legend:
OP Operation Code (Hexadecimal);
~ Number of MPU Cycles;
# Number of Program Bytes;
+ Arithmetic Plus;
− Arithmetic Minus;
• Boolean AND;
M_{SP} Contents of memory location pointed to be Stack Pointer;
+ Boolean Inclusive OR;
⊕ Boolean Exclusive OR;
M Complement of M;
→ Transfer Into;
0 Bit = Zero;
00 Byte = Zero;

Condition Code Symbols:
H Half-carry from bit 3;
I Interrupt mask
N Negative (sign bit)
Z Zero (byte)
V Overflow, 2's complement
C Carry from bit 7
R Reset Always
S Set Always
↕ Test and set if true, cleared otherwise
• Not Affected

Program Control Operations

Program Control operation can be subdivided into two categories: (1) index register/stack pointer instructions; (2) jump and branch operations.

Index Register/Stack Pointer Operations

The instructions for direct operation on the MPU's index register and stack pointer are summarized in *Table 5*. Decrement (DEX, DES), increment (INX, INS), load (LDX, LDS), and store (STX, STS) instructions are provided for both. The compare instruction, CPX, can be used to compare the index register to a 16-bit value and update the condition code register accordingly.

The TSX instruction causes the index register to be loaded with the address of the last data byte put onto the stack. The TXS instruction loads the stack pointer with a value equal to one less than the current contents of the index register. This causes the next byte to be pulled from the stack to come from the location indicated by the index register. The utility of these two instructions can be clarified by describing the stack concept relative to the F6800 system.

The stack can be thought of as a sequential list of data stored in the MPU's read/write memory. The stack pointer contains a 16-bit memory address that is used to access the list from one end on a last-in-first-out (LIFO) basis in contrast to the random access mode used by the MPU's other addressing modes.

The F6800 instruction set and interrupt structure allow extensive use of the stack concept for efficient handling of data movement, subroutines and interrupts. The instructions can be used to establish one or more stacks anywhere in read/write memory. Stack length is limited only by the amount of memory that is made available.

Table 5 Index Register and Stack Pointer Instructions

Pointer Operations	Mnemonic	Immed			Direct			Index			Extend			Implied			Boolean/Arithmetic Operation	Cond. Code Reg.*					
																		5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M + 1)$	•	•	⑦	↕	⑧	•
Decrement Index Reg	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M + 1) \rightarrow X_L$	•	•	⑨	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M + 1) \rightarrow SP_L$	•	•	⑨	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M + 1)$	•	•	⑨	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M + 1)$	•	•	⑨	↕	R	•
Indx Reg → Stack Pntr	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Stack Pntr → Indx Reg	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

*See condition code register notes page 26

Operation of the stack pointer with the push and pull instructions is illustrated in *Figures 8* and *9*. The push instruction (PSHA) causes the contents of the indicated accumulator (A in this example) to be stored in memory at the location indicated by the stack pointer. The stack pointer is automatically decremented by one following the storage operation and is "pointing" to the next empty stack location. The pull instruction (PULA or PULB) causes the last byte stacked to be loaded into the appropriate accumulator. The stack pointer is automatically incremented by one just prior to the data transfer so that it will point to the last byte stacked rather than the next empty location. Note that the pull instruction does not remove the data from memory; in the example, 1A is still in location (m + 1) following execution of PULA. A subsequent push instruction would overwrite that location with the new pushed data.

Execution of the branch to subroutine (BSR) and jump to subroutine (JSR) instructions cause a return address to be saved on the stack as shown in *Figures 11* through *13*. The stack is decremented after each byte of the return address is pushed onto the stack. For both of these instructions, the return address is the memory location following the bytes of code that correspond to the BSR and JSR instruction. The code required for BSR or JSR may be either two or three bytes, depending on whether the JSR is in the indexed (two bytes) or the extended (three bytes) addressing mode. Before it is stacked, the program counter is automatically incremented the correct number of times to be pointing at the location of the next instruction. The return from subroutine instruction, RTS, causes the return address to be retrieved and loaded into the program counter as shown in *Figure 14*.

There are several operations that cause the status of the MPU to be saved on the stack. The software interrupt (SWI) and wait for interrupt (WAI) instructions as well as the maskable ($\overline{IRQ}$) and non-maskable ($\overline{NMI}$) hardware interrupts all cause the MPU's internal registers (except for the stack pointer itself) to be stacked as shown in *Figure 16*. MPU status is restored by the return from interrupt, RTI, as shown in *Figure 15*.

Fig. 8 Stack Operation, Push Instruction

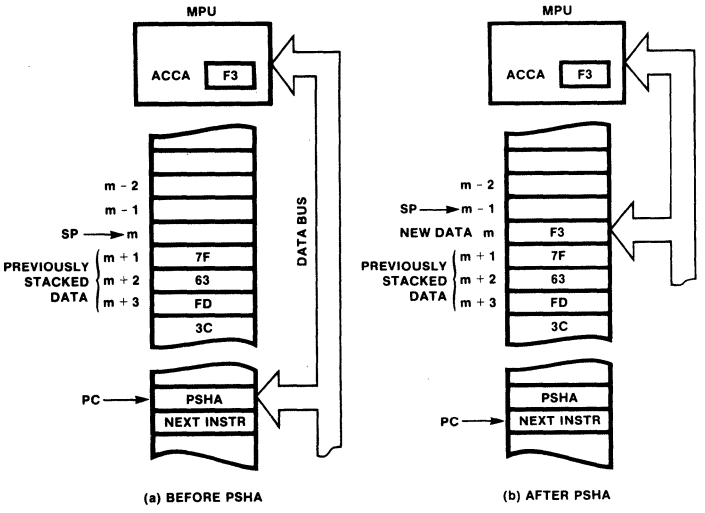


Fig. 9 Stack Operation, Pull Instruction

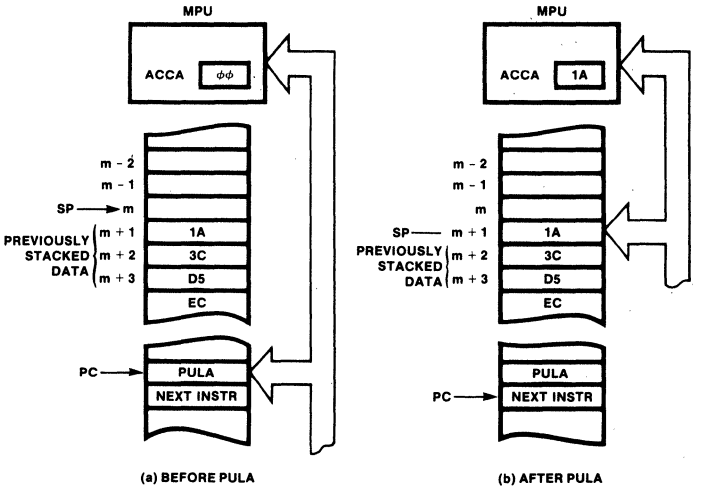


Fig. 10 Program Flow for Jump and Branch Instructions

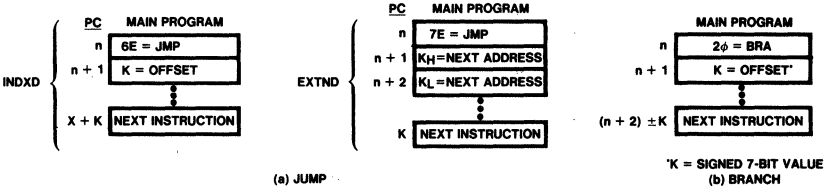


Fig. 11 Program Flow for BSR

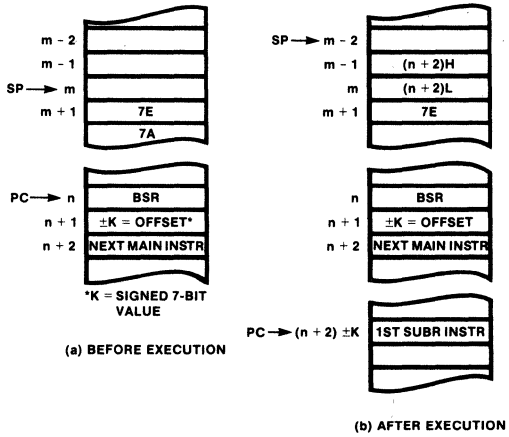


Fig. 12 Program Flow for JSR (Extended)

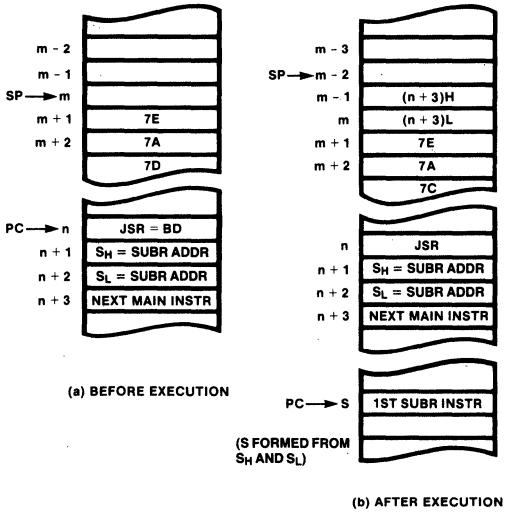


Fig. 13 Program Flow for JSR (Indexed)

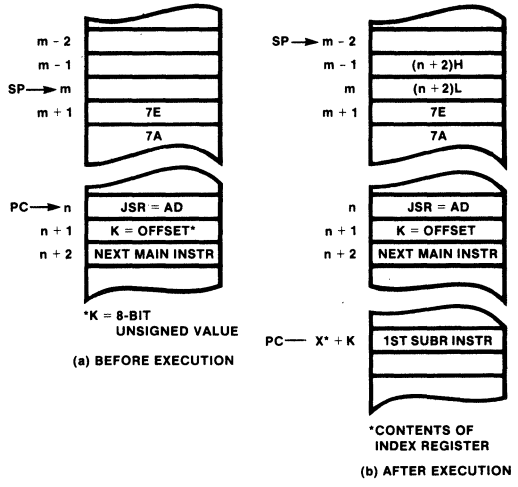


Fig. 14 Program Flow for RTS

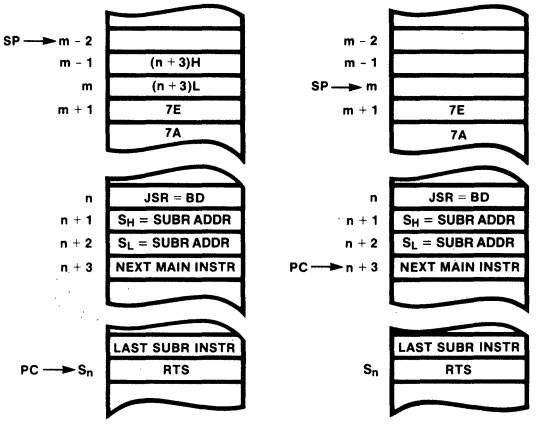


Fig. 15 Program Flow for RTI

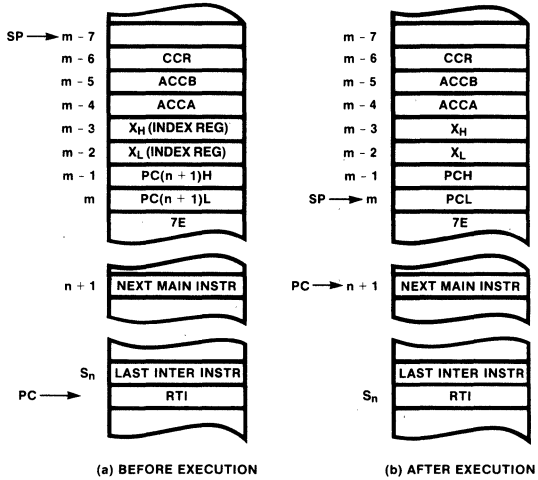
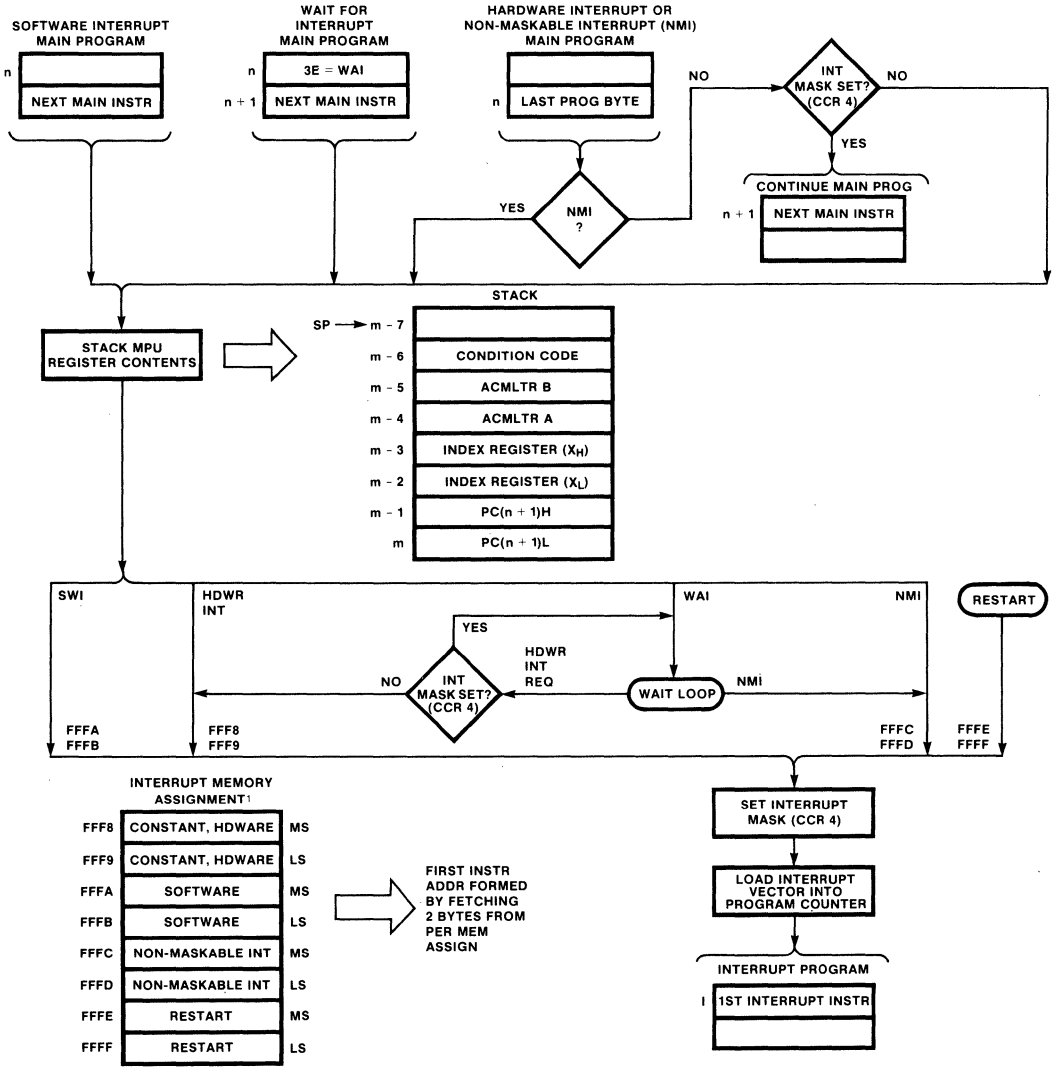


Fig. 16 Program Flow for Interrupts



Note
MS = Most Significant Address Byte
LS = Least Significant Address Byte

Jump and Branch Operation

The jump and branch instructions are summarized in *Table 6*. These instructions are used to control the transfer of operation from one point to another in the control program.

The no operation instruction, NOP, while included here, is a jump operation in a very limited sense. Its only effect is to increment the program counter by one. It is useful during program development as a stand-in for some other instruction that is to be determined during debug. It is also used for equalizing the execution time through alternate paths in a control program.

Execution of the jump instruction, JMP, and branch always, BRA, affects program flow as shown in *Figure 10*. When the MPU encounters the jump (indexed) instruction, it adds the offset to the value in the index register and uses the result as the address of the next instruction to be executed. In the extended addressing mode, the address of the next instruction to be executed is fetched from the two locations immediately following the JMP instruction. The branch always (BRA) instruction is similar to the JMP (extended) instruction except that the relative addressing mode applies and the branch is limited to the range within −125 or +127 bytes of the branch instruction itself. The opcode for the BRA instruction requires one less byte than JMP (extended) but takes one more cycle to execute.

Table 6 Jump and Branch Instructions

Operations	Mnemonic	Relative			Index			Extend			Implied			Branch Test	Cond. Code Reg. †					
		OP	~	#	OP	~	#	OP	~	#	OP	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•
Branch if Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•
Branch if Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•
Branch if = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
Branch if ≥ Zero	BGE	2C	4	2										N + V = 0	•	•	•	•	•	•
Branch if > Zero	BGT	2E	4	2										Z + (N + V) = 0	•	•	•	•	•	•
Branch if Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
Branch if ≤ Zero	BLE	2F	4	2										Z + (N + V) = 1	•	•	•	•	•	•
Branch if Lower or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
Branch if < Zero	BLT	2D	4	2										N + V = 1	•	•	•	•	•	•
Branch if Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Branch if not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Branch if Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•
Branch if Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•
Branch if Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Branch to Subroutine	BSR	8D	8	2											•	•	•	•	•	•
Jump	JMP				6E	4	2	7E	3	3				See Special Operations	•	•	•	•	•	•
Jump to Subroutine	JSR				AD	8	2	BD	9	3				See Special Operations	•	•	•	•	•	•
No Operation	NOP										01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return from Interrupt	RTI										3B	10	1		———			⑩		———
Return from Subroutine	RTS										39	5	1	See Special Operations	•	•	•	•	•	•
Software Interrupt	SWI										3F	12	1	See Special Operations	•	•	•	•	•	•
Wait for Interrupt*	WAI										3E	9	1	See Special Operations	•	⑪	•	•	•	•

*WAI puts address bus, R/$\overline{W}$, and data bus in the 3-state mode while VMA is held LOW.
†See condition code register notes page 26.

The effect on program flow for the jump to subroutine (JSR) and branch to subroutine (BSR) is shown in *Figures 11* through *13*. Note that the program counter is properly incremented to be pointing at the correct return address before it is stacked. Operation of the branch to subroutine and jump to subroutine (extended) instruction is similar except for the range. The BSR instruction requires less opcode than JSR (2 bytes versus 3 bytes) and also executes one cycle faster than JSR. The return from subroutine, RTS, is used at the end of a subroutine to return to the main program as indicated in *Figure 14*.

The effect of executing the software interrupt, SWI, and the wait for interrupt, WAI, and their relationship to the hardware interrupts is shown in *Figure 15*. SWI causes the MPU contents to be stacked and then fetches the starting address of the interrupt routine from the memory locations that respond to the addresses FFFA and FFFB. Note that as in the case of the subroutine instructions, the program counter is incremented to point at the correct return address before being stacked. The return from interrupt instruction, RTI, (*Figure 15*) is used at the end of an interrupt routine to restore control to the main program. The SWI instruction is useful for inserting break points in the control program, that is, it can be used to stop operation and put the MPU registers in memory where they can be examined. The WAI instruction is used to decrease the time required to service a hardware interrupt; it stacks the MPU contents and then waits for the interrupt to occur, effectively removing the stacking time from a hardware interrupt sequence.

Fig. 17 Conditional Branch Instructions

BMI : $N = 1$;	BEQ : $Z = 1$;
BPL : $N = \phi$;	BNE : $Z = \phi$;
BVC : $V = \phi$;	BCC : $C = \phi$;
BVS : $V = 1$;	BCS : $C=1$;
BHI : $C + Z = \phi$;	BLT : $N \oplus V = 1$;
BLS : $C + Z = 1$;	BGE : $N \oplus V = \phi$;
BLE : $Z + (N \oplus V) = 1$;	
BGT : $Z + (N \oplus V) = \phi$;	

The conditional branch instructions, *Figure 17*, consist of seven pairs of complementary instructions. They are used to test the results of the preceding operation and either continue with the next instruction in sequence (test fails), or cause a branch to another point in the program (test succeeds).

Four of the pairs are used for simple tests of status bits N, Z, V, and C:

1. Branch on minus (BMI) and branch on plus (BPL) tests the sign bit, N, to determine if the previous result was negative or positive, respectively.
2. Branch on equal (BEQ) and branch on not equal (BNE) are used to test the zero status bit, Z, to determine whether or not the result of the previous operation was equal to zero. These two instructions are useful following a compare (CMP) instruction to test for equality between an accumulator and the operand. They are also used following the bit test (BIT) to determine whether or not the same bit positions are set in an accumulator and the operand.
3. Branch on overflow clear (BVC) and branch on overflow set (BVS) tests the state of the V bit to determine if the previous operation caused an arithmetic overflow.
4. Branch on carry clear (BCC) and branch on carry set (BCS) tests the state of the C bit to determine if the previous operation caused a carry to occur. BCC and BCS are useful for testing relative magnitude when the values being tested are regarded as unsigned binary numbers, that is, the values are in the range 00 (lowest) to FF (highest). BCC following a comparison (CMP) will cause a branch if the (unsigned) value in the accumulator is higher than or the same as the value of the operand. Conversely, BCS will cause a branch if the accumulator value is lower than the operand.

The fifth complementary pair, branch on higher (BHI) and branch on lower or same (BLS) are in a sense complements to BCC and BCS. BHI tests for both C and Z = 0; if used following a CMP, it will cause a branch if the value in the accumulator is higher than the operand. Conversely, BLS will cause a branch if the unsigned binary value in the accumulator is lower than or the same as the operand.

The remaining two pairs are useful in testing results of operations in which the values are regarded as signed two's complement numbers. This differs from the unsigned binary case in the following sense: In unsigned, the orientation is higher or lower; in signed two's complement, the comparison is between larger or smaller where the range of values is between −128 and +127.

Branch on less than zero (BLT) and branch on greater than or equal zero (BGE) test the status bits for $N \oplus V$ = "1" and $N \oplus V$ = "0", respectively. BLT will always cause a branch following an operation in which two negative numbers were added. In addition, it will cause a branch following a CMP in which the value in the accumulator was negative and the operand was positive. BLT will never cause a branch following a CMP in which the accumulator value was positive and the operand negative. BGE, the complement to BLT, will cause a branch following operations in which two positive values were added or in which the result was zero.

The last pair, branch on less than or equal zero (BLE) and

branch on greater than zero (BGT) test the status bits for Z $\oplus$ (N + V) = "1" and Z $\oplus$ (N + V) = "0", respectively. The action of BLE is identical to that for BLT except that a branch will also occur if the result of the previous result was zero. Conversely, BGT is similar to BGE except that no branch will occur following a zero result.

Condition Code Register Operations

The condition code register (CCR) is a 6-bit register within the MPU that is useful in controlling program flow during system operation. The bits are defined in *Figure 18.*

The instructions shown in *Table 7* are available to the user for direct manipulation of the CCR. In addition, the MPU automatically sets or clears the appropriate status bits as many of the other instructions on the condition code register were indicated as they were introduced.

Table 7 Condition Code Register Instructions

Operations	Mnemonic	Implied			Boolean Operation	Cond. Code Reg.*					
						5	4	3	2	1	0
		OP	~	#		H	I	N	Z	V	C
Clear Carry	CLC	OC	2	1	0 → C	●	●	●	●	●	R
Clear Interrupt Mask	CLI	OE	2	1	0 → I	●	R	●	●	●	●
Clear Overflow	CLV	OA	2	1	0 → V	●	●	●	●	R	●
Set Carry	SEC	OD	2	1	1 → C	●	●	●	●	●	S
Set Interrupt Mask	SEI	OF	2	1	1 → I	●	S	●	●	●	●
Set Overflow	SEV	OB	2	1	1 → V	●	●	●	●	S	●
Acmltr A → CCR	TAP	06	2	1	A → CCR	(12)					
CCR → Acmltr A	TPA	07	2	1	CCR → A	●	●	●	●	●	●

R = Reset
S = Set
● = Not affected
1 (ALL) Set according to the contents of Accumulator A.
*See Condition Code Register notes below

Condition Code Register Notes: (Bit set if test is true and cleared otherwise)

1 (Bit V) Test: Result = 10000000?

2 (Bit C) Test: Result = 00000000?

3 (Bit C) Test: Decimal value of most significant BCD character greater than nine? (Not cleared if previously set.)

4 (Bit V) Test: Operand = 10000000 prior to execution?

5 (Bit V) Test: Operand = 01111111 prior to execution?

6 (Bit V) Test: Set equal to result of N $\oplus$ C after shift has occurred.

7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?

8 (Bit V) Test: 2s complement overflow from subtraction of MS bytes?

9 (Bit N) Test: Result less than "0"? (Bit 15 = 1)

10 (All) Load condition code register from stack. (See Special Operations)

11 (Bit I) Set when interrupt occurs. If previously set, a non-maskable interrupt is required to exit the wait state.

12 (All) Set according to the contents of accumulator A.

Fig. 18 Condition Code Register Bit Definition

b_5	b_4	b_3	b_2	b_1	b_0
H	I	N	Z	V	C

H = Half-carry; set whenever a carry from b_3 to b_4 of the result is generated by ADD, ABA, ADC; cleared if no b_3 to b_4 carry; not affected by other instructions.

I = Interrupt Mask; set by hardware or software interrupt or SEI instruction; cleared by CLI instruction. (Normally not used in arithmetic operations.) Restored to a zero as a result of an RTI instruction if I_m stored on the stack is LOW.

N = Negative; set if high order bit (b_7) of result is set; cleared otherwise

Z = Zero; set if result = 0; cleared otherwise.

V = Overlow; set if there were arithmetic overflow as a result of the operation; cleared otherwise.

C = Carry; set if there were a carry from the most significant bit (b_7) of the result; cleared otherwise.

A CLI-WAI instruction sequence operated properly with early F6800 processors only if the preceding instruction were odd. (Least Significant Bit = "1".) Similarly it was advisable to precede any SEI instruction with an odd opcode—such as NOP. These precautions are not necessary for F6800 processors indicating manufacture in November, 1977 or later.

Systems which require an interrupt window to be opened under program control should use a CLI-NOP-SEI sequence rather than CLI-SEI.

Addressing Modes

The MPU operates on 8-bit binary numbers presented to it via the data bus. A given number (byte) may represent either data or an instruction to be executed, depending on where it is encountered in the control program. The F6800 has 72 unique instructions; however, it recognizes and takes action on 197 of the 256 possibilities that can occur using an 8-bit word length. This larger number of instructions results from the fact that many of the executive instructions have more than one addressing mode.

These addressing modes refer to the manner in which the program causes the MPU to obtain its instructions and data. The programmer must have a method for addressing the MPU's internal registers and all of the external memory locations.

Selection of the desired addressing mode is made by the user as the source statements are written. Translation into appropriate opcode then depends on the method used. If manual translation is used, the addressing mode is inherent in the opcode. For example, the immediate, direct, indexed, and extended modes may all be used with the ADD instruction. The proper mode is determined by selecting (hexadecimal notation) 8B, 9B, AB, or BB, respectively.

The source statement format includes adequate information for the selection if an assembler program is used to generate the opcode. For instance, the immediate mode is selected by the assembler whenever it encounters the "#" symbol in the operand field. Similarly, an "X" in the operand field causes the indexed mode to be selected. Only the relative mode applies to the branch instructions; therefore, the mnemonic instruction itself is enough for the assembler to determine addressing mode.

5

For the instructions that use both direct and extended modes, the assembler selects the direct mode if the operand value is in the range 0-255 and extended otherwise. There are a number of instructions for which the extended mode is valid but the direct is not. For these instructions, the assembler automatically selects the extended mode even if the operand is in the 0-255 range. The addressing modes are summarized in *Figure 19.*

Inherent (Includes "Accumulator Addressing") Mode

The successive fields in a statement are normally separated by one or more spaces. An exception to this rule occurs for instructions that use dual addressing in the operand field and for instructions that must distinguish between the two accumulators. In these cases, A and B are "operands" but the space between them and the operator may be omitted. This is commonly done, resulting in apparent four character mnemonics for those instructions.

The addition instruction, ADD, provides an example of dual addressing in the operand field:

	Operator	Operand	Comment
	ADDA	MEM12	ADD CONTENTS OF MEM12 TO ACCA
or	ADDB	MEM12	ADD CONTENTS OF MEM12 TO ACCB

The example used earlier for the test instruction, TST, also applies to the accumulators and uses the "accumulator addressing mode" to designate which of the two accumulators is being tested:

	Operator	Comment
	TSTB	TEST CONTENTS OF ACCB
or	TSTA	TEST CONTENTS OF ACCA

A number of the instructions either alone or together with an accumulator operand contain all of the address information that is required, that is, "inherent" in the instruction itself. For instance, the instruction ABA causes the MPU to add the contents of accumulators A and B together and place the result in accumulator A. The instruction INCB, another example of "accumulator addressing", causes the contents of accumulator B to be increased by one. Similarly, INX, incrementing the index register, causes the contents of the index register to be increased by one.

Program flow for instructions of this type is illustrated in *Figures 20* and *21*. In these figures, the general case is shown on the left and a specific example is shown on the right. Numerical examples are in decimal notation. Instructions of this type require only one byte of opcode. Cycle-by-cycle operation of the inherent mode is shown in *Table 8*.

Direct and Extended Addressing Modes

In the direct and extended modes of addressing, the operand field of the source statement is the *address* of the value that is to be operated on. The direct and extended modes differ only in the range of memory locations to which they can direct the MPU. Direct addressing generates a single 8-bit operand and, hence, can address only memory locations 0 through 255; a two byte operand is generated for extended addressing, enabling the MPU to reach the remaining memory locations, 256 through 65535. An example of direct addressing and its effect on program flow is illustrated in *Figure 23*.

The MPU, after encountering the opcode for the instruction LDAA (direct) at memory location 5004 (program counter = 5004), looks in the next location, 5005, for the address of the operand. It then sets the program counter equal to the

Fig. 19 Addressing Mode Summary

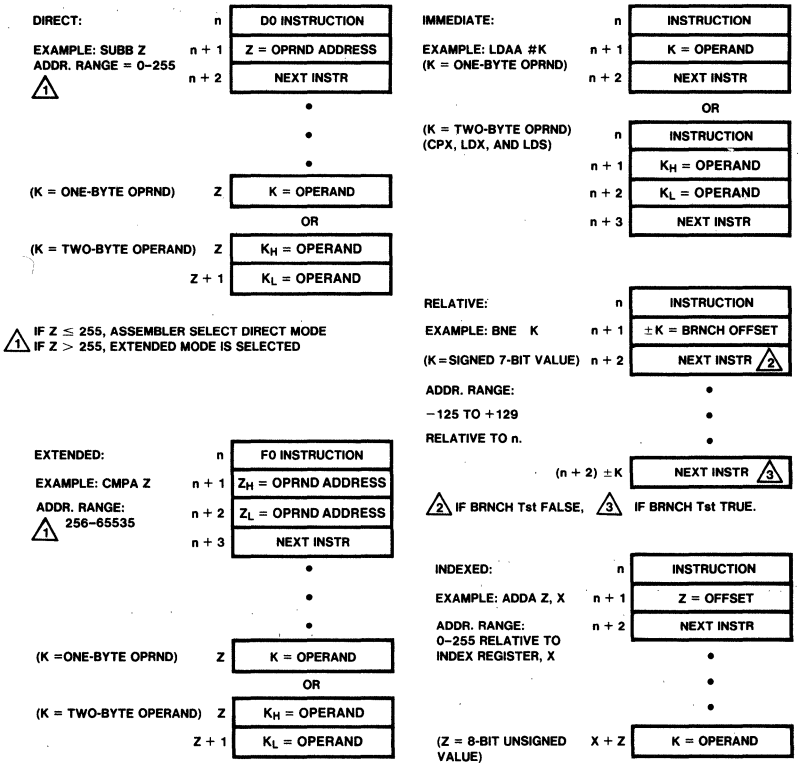


Table 8 Inherent Mode Cycle-by-Cycle Operation

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Inherent						
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	1 1	Op Code Address Op Code Address + 1	1 1	Op Code Op Code of Next Instruction
DES DEX INS INX	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Previous Register Contents New Register Contents	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data (Note 1) Irrelevant Data (Note 1)
PSH	4	1 2 3 4	1 1 1 0	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer − 1	1 1 0 1	Op Code Op Code of Next Instruction Accumulator Data Accumulator Data
PUL	4	1 2 3 4	1 1 0 1	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data (Note 1) Operand Data from Stack
TSX	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Stack Pointer New Index Register	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data (Note 1) Irrelevant Data (Note 1)
TXS	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Index Register New Stack Pointer	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data Irrelevant Data
RTS	5	1 2 3 4 5	1 1 0 1 1	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Op Code Irrelevant Data (Note 2) Irrelevant Data (Note 1) Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)

Table 8 Inherent Mode Cycle-by-Cycle Operation (Cont.)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Inherent (Cont'd)						
WAI	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer − 2	0	Index Register (Low Order Byte
		6	1	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer − 4	0	Contents of Accumulator A
		8	1	Stack Pointer − 5	0	Contents of Accumulator B
		9	1	Stack Pointer − 6 (Note 3)	1	Contents of Cond. Code Register
RTI	10	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 2)
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Contents of Cond. Code Register from Stack
		5	1	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	1	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	1	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	1	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	1	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	1	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 1)
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer − 4	0	Contents of Accumulator A
		8	1	Stack Pointer − 5	0	Contents of Accumulator B
		9	1	Stack Pointer − 6	0	Contents of Cond. Code Register
		10	0	Stack Pointer − 7	1	Irrelevant Data (Note 1)
		11	1	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	1	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

Notes

1. If device which is addressed during this cycle uses VMA, then the data bus will go to the high impedance 3-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the data bus.
2. Data is ignored by the MPU.
3. While the MPU is waiting for the interrupt, Bus Available will go HIGH indicating the following states of the control lines: VMA is LOW; address bus, R/$\overline{W}$, and data bus are all in the high impedance state.

Fig. 20 Inherent Addressing

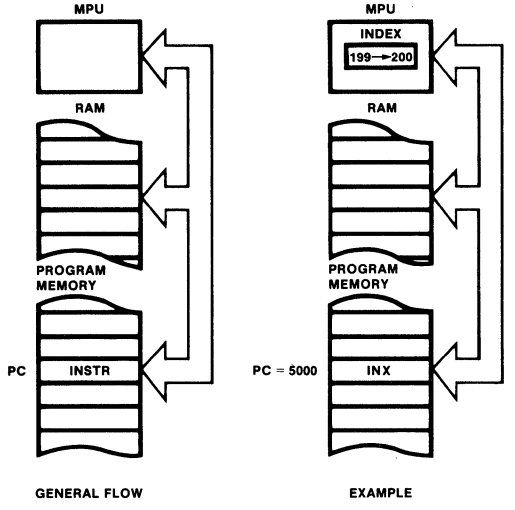


Fig. 21 Accumulator Addressing

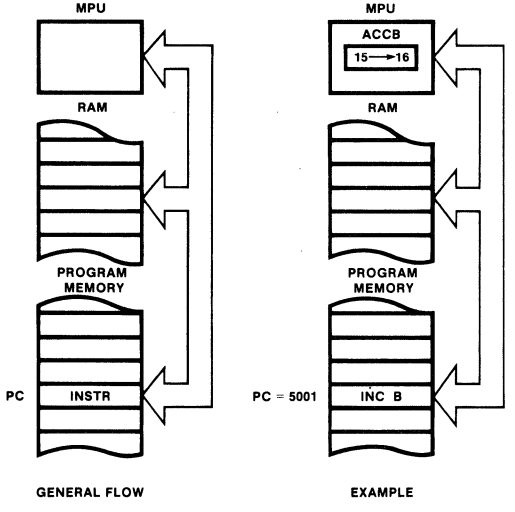


Fig. 22 Immediate Addressing Mode

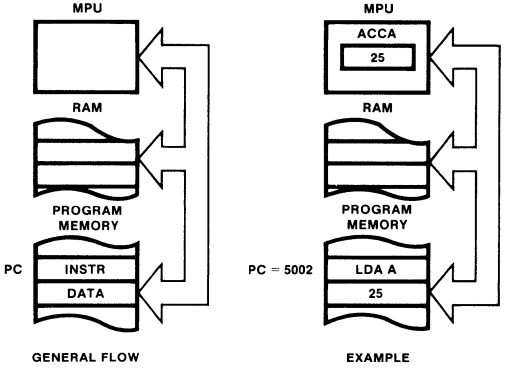
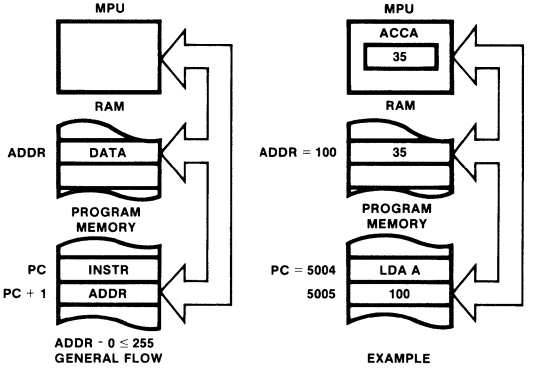


Fig. 23 Direct Addressing Mode



5

Table 9 Immediate Mode Cycle-by-Cycle Operation

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Immediate						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	1 1	Op Code Address Op Code Address +1	1 1	Op Code Operand Data
CPX LDS LDX	3	1 2 3	1 1 1	Op Code Address Op Code Address +1 Op Code address +2	1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte)

Table 10 Direct Mode Cycle-by-Cycle Operation

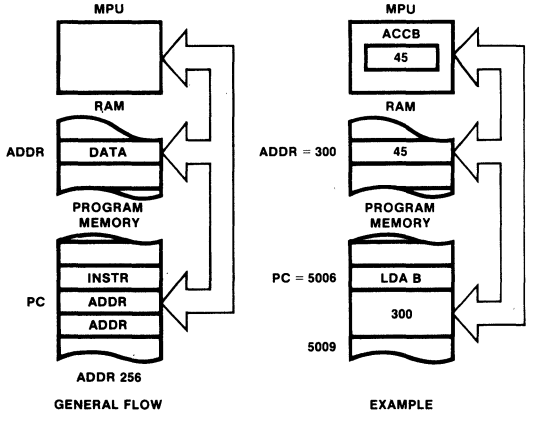
Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Direct						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	1 1 1	Op Code Address Op Code Address +1 Address of Operand	1 1 1	Op Code Address of Operand Operand Data
CPX LDS LDX	4	1 2 3 4	1 1 1 1	Op Code Address Op Code Address +1 Address of Operand Operand Address +1	1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STA	4	1 2 3 4	1 1 0 1	Op Code Address Op Code Address +1 Destination Address Destination Address	1 1 1 0	Op Code Destination Address Irrelevant Data (Note) Data from Accumulator
STS STX	5	1 2 3 4 5	1 1 0 1 1	Op Code Address Op Code Address +1 Address of Operand Address of Operand Address of Operand +1	1 1 1 0 0	Op Code Address of Operand Irrelevant Data (Note) Register Data (High Order Byte) Register Data (Low Order Byte)

Note
If device which is addressed during this cycle uses VMA, then the data bus will go to the high impedance 3-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the data bus.

value found there (100 in the example) and fetches the operand, in this case a value to be loaded into accumulator A, from that location. For instructions requiring a 2-byte operand such as LDX (load the index register), the operand bytes would be retrieved from locations 100 and 101. *Table 10* shows the cycle-by-cycle operations for the direct mode of addressing.

Extended addressing, *Figure 24*, is similar except that a two-byte address is obtained from locations 5007 and 5008 after the LDAB (extended) opcode shows up in location 5006. Extended addressing can be thought of as the standard addressing mode, that is, it is a method of reaching any place in memory. Direct addressing, since only one address byte is required, provides a faster method of processing data and generates fewer bytes of control code. In most applications, the direct addressing range, memory locations 0-255, are reserved for RAM. They are used for data buffering and temporary storage of system variables, the area in which faster addressing is of most value. Cycle-by-cycle operation is shown in *Table 11* for extended addressing.

Fig. 24 Extended Addressing Mode



Immediate Addressing Mode

In the immediate addressing mode, the operand is the value that is to be operated on. For instance, the instruction

Operator	Operand	Comment
LDAA	#25	LOAD 25 INTO ACCA

causes the MPU to "immediately load accumulator A with the value 25"; no further address reference is required. The immediate mode is selected by preceding the operand value with the "#" symbol. Program flow for this addressing mode is illustrated in *Figure 22*.

The operand format allows either properly defined symbols or numerical values. Except for the instructions CPX, LDX, and LDS, the operand may be any value in the range 0 to 255. Since compare index register (CPX), load index register (LDX), and load stack pointer (LDS), require 16-bit values, the immediate mode for these three instructions requires two-byte operands. In the immediate addressing mode, the "address" of the operand is effectively the memory location immediately following the instruction itself. *Table 9* shows the cycle-by-cycle operation for the immediate addressing mode.

Relative Addressing Mode

In both the direct and extended modes, the address obtained by the MPU is an absolute numerical address. The relative addressing mode, implemented for the MPU's branch instructions, specifies a memory location relative to the program counter's current location. Branch instructions generate two bytes of machine code, one for the instruction opcode and one for the "relative" address. (See *Figure 25.*) Since it is desirable to be able to branch in either direction, the 8-bit address byte is interpreted as a signed 7-bit value; the 8th bit of the operand is treated as a sign bit, "0" = plus and "1" = minus. The remaining seven bits represent the numerical value. This results in a relative addressing range of ± 127 with respect to the location of the branch instruction itself. However, the branch range is computed with respect to the next instruction that would be executed if the branch conditions are not satisfied. Since two bytes are generated, the next instruction is located at PC + 2. If D is defined as the address of the branch designation, the range is then:

$$(PC + 2) - 127 \leq D \leq (PC + 2) + 127$$

or $$PC - 125 \leq D \leq PC + 129$$

that is, the destination of the branch instruction must be within −125 to +129 memory locations of the branch instructions itself. For transferrring control beyond this range, the unconditional jump (JMP), jump to subroutine (JSR), and return from subroutine (RTS) are used.

In *Figure 25,* when the MPU encounters the opcode for BEQ (branch if result of last instruction was zero), it tests the zero bit in the condition code register. If that bit is "0", indicating a non-zero result, the MPU continues execution with the next instruction (in location 5010 in *Figure 25*). If the previous result were zero, the branch condition is satisfied and the MPU adds the offset, 15 in this case, to PC + 2 and branches to location 5025 for the next instruction.

Table 11 Extended Mode Cycle-by-Cycle Operation

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Extended						
STS STX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	0	Address of Operand	1	Irrelavant Data (Note 1)
		5	1	Address of Operand	0	Operand Data (High Order Byte)
		6	1	Address of Operand + 1	0	Operand Data (Low Order Byte)
JSR	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	1	Subroutine Starting Address	1	Op Code of Next Instruction
		5	1	Stack Pointer	0	Return Address (Low Order Byte)
		6	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		7	0	Stack Pointer − 2	1	Irrelevant Data (Note 1)
		8	0	Op Code Address + 2	1	Irrelevant Data (Note 1)
		9	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
JMP	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	1	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data
CPX LDS LDX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data (High Order Byte)
		5	1	Address of Operand + 1	1	Operand Data (Low Order Byte)
STA A STA B	5		1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	1	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	0	Operand Destination Address	1	Irrelevant Data (Note 1)
		5	1	Operand Destination Address	0	Data from Accumulator
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Curent Operand Data
		5	0	Address of Operand	1	Irrelevant Data (Note 1)
		6	1/0 (Note 2)	Address of Operand	0	New Operand Data (Note 2)

Notes

1. If device which is addressed during this cycle uses VMA, then the data bus will go to the high impedance 3-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the data bus.
2. For TST, VMA = "0" and operand data does not change.

The branch instructions allow the programmer to efficiently direct the MPU to one point or another in the control program depending on the outcome of test results. Since the control program is normally in read-only memory and cannot be changed, the relative address used in execution of branch instructions is a constant numerical value. Cycle-by-cycle operation is shown in *Table 12* for relative addressing.

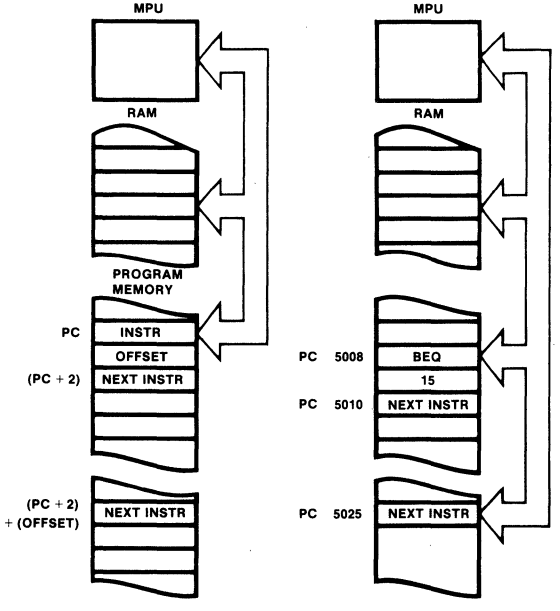
Indexed Addressing Mode

With indexed addressing, the numerical address is variable and depends on the current contents of the index register. A source statement such as

Operator	Operand	Comment
STAA	X	Put A in Indexed Location

causes the MPU to store the contents of accumulator A in the memory location specified by the contents of the index register (recall that the label "X" is reserved to designate the index register). Since there are instructions for manipulating X during program execution (LDX, INX, DEX, etc.), the indexed addressing mode provides a dynamic on-the-fly way to modify program activity.

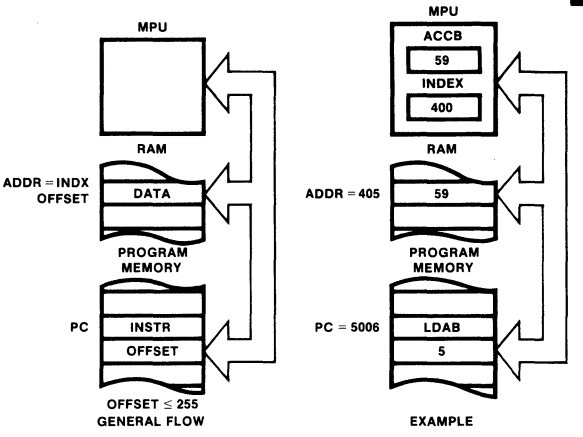
Fig. 25 Relative Addressing Mode



The operand field can also contain a numerical value that will be automatically added to X during execution. This format is illustrated in *Figure 26*.

When the MPU encounters the LDAB (Indexed) opcode in location 5006, it looks in the next memory location for the value to be added to X (5 in the example) and calculates the required address by adding 5 to the present index register value of 400. In the operand format, the offset may be represented by a label or a numerical value in the range 0-255 as in the example. In the earlier example, STAA X, the operand is equivalent to 0, X, that is, the 0 may be omitted when the desired address is equal to X. *Table 13* shows the cycle-by-cycle operation for the indexed mode of addressing.

Fig. 26 Indexed Addressing Mode



5

Table 12 Relative Mode Cycle-by-Cycle Operation

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Relative						
BCC BHI BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMI BVS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Op Code Address + 2	1	Irrelevant Data (Note)
		4	0	Branch Address	1	Irrelevant Data (Note)
BSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Return Address of Main Program	1	Irrelevant Data (Note)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer − 2	1	Irrelevant Data (Note)
		7	0	Return Address of Main Program	1	Irrelevant Data (Note)
		8	0	Subroutine Address	1	Irrelevant Data (Note)

Note
If device which is addressed during this cycle uses VMA, then the data bus will go to the high impedance 3-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the data bus.

Table 13 Indexed Mode Cycle-by-Cycle Operation

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Indexed						
JMP	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Operand Data
CPX LDS LDX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Operand Data (High Order Byte)
		6	1	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STA	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data

Table 13 Indexed Mode Cycle-by-Cycle Operation (Cont.)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
Indexed (Cont.)						
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Current Operand Data
		6	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		7	1/0 (Note 2)	Index Register Plus Offset	0	New Operand Data (Note 2)
STS STX	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data (High Order Byte)
		7	1	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
JSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer − 2	1	Irrelevant Data (Note 1)
		7	0	Index Register	1	Irrelevant Data (Note 1)
		8	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)

Note

1. If device which is addressed during this cycle uses VMA, then the data bus will go to the high impedance 3-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the data bus.

2. For TST, VMA = "0" and operand data does not change.

Absolute Maximum Ratings

Supply Voltage	−0.3 V, +7.0 V
Input Voltage	−0.3 V, +7.0 V
Operating Temperature Range—T_L to T_H	
F6800, F68A00, F68B00	0°C, +70°C
F6800C, F68A00C, F68B00C	−40°C, +85°C
F6800DM, F68A00DM, F68B00DM	−55°C, +125°C
Storage Temperature Range	−55°C, +150°C
Thermal Resistance	
Plastic Package	70°C/W
Ceramic Package	50°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit.

DC Characteristics V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted

Symbol	Characteristic		Min	Typ	Max	Unit	Conditions
V_{IH} V_{IHC}	Input HIGH Voltage	Logic $\phi1$, $\phi2$	V_{SS} + 2.0 V_{CC} − 0.6		V_{CC} V_{CC} + 0.3	V	
V_{IL} V_{ILC}	Input LOW Voltage	Logic $\phi1$, $\phi2$	V_{SS} − 0.3 V_{SS} − 0.3		V_{SS} + 0.8 V_{SS} + 0.4	V	
I_{IN}	Input Leakage Current	Logic $\phi1$, $\phi2$		1.0	2.5 100	μA	V_{IN} = 0 to 5.25 V, V_{CC} = Max V_{IN} = 0 to 5.25 V, V_{CC} = 0.0 V
I_{TSI}	3-State (OFF State) Input Current	D_0-D_7 A_0-A_{15}, R/$\overline{W}$		2.0	10 100	μA	V_{IN} = 0.4 to 2.4 V, V_{CC} = Max
V_{OH}	Output HIGH Voltage	D_0-D_7 A_0-A_{15}, R/$\overline{W}$, VMA BA	V_{SS} + 2.4 V_{SS} + 2.4 V_{SS} + 2.4			V	I_{Load} = −205 μA, V_{CC} = Min I_{Load} = −145 μA, V_{CC} = Min I_{Load} = −100 μA, V_{CC} = Min
V_{OL}	Output LOW Voltage				V_{SS} + 0.4	V	I_{Load} = 1.6 mA, V_{CC} = Min
P_D	Power Dissipation			0.5	1.0	W	
C_{IN}	Input Capacitance	$\phi1$ $\phi2$ D_0-D_7 Logic Inputs		25 45 10 6.5	35 70 12.5 10	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance A_0-A_{15}, R/$\overline{W}$, VMA				12	pF	

Clock Timing V_{CC} = 5.0 V ± 5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted

Symbol	Characteristic		Min	Typ	Max	Unit	Conditions
f	Frequency of Operation						
		F6800	0.1		1.0	MHz	
		F68A00	0.1		1.5		
		F68B00	0.1		2.0		
t_{cyc}	Cycle Time *(Figure 27)*	F6800	1.000		10	μs	
		F68A00	0.666		10		
		F68B00	0.500		10		
$PW_{\phi H}$	Clock Pulse Width						
	$\phi1$, $\phi2$ -	F6800	400		9500	ns	V_{CC} − 0.6 V
	$\phi1$, $\phi2$ -	F68A00	230		9500		
	$\phi1$, $\phi2$ -	F68B00	180		9500		
t_{ut}	Total $\phi1$ and $\phi2$ Up Time	F6800	900			ns	
		F68A00	600				
		F68B00	440				
$t_{\phi r}$, $t_{\phi f}$	Rise and Fall Times				100	ns	Measured Between V_{SS} + 0.4 V and V_{CC} − 0.6 V
t_d	Delay Time or Clock Separation *(Figure 27)*		0 0		9100 9100	ns	$V_{OV} = V_{SS} + 0.6$ V @ $t_r = t_f \leq 100$ ns $V_{OV} = V_{SS} + 1.0$ V @ $t_r = t_f \leq 35$ ns

Read/Write Timing (Reference *Figures 28* through *32*)

Symbol	Characteristic	F6800			F68A00			F68B00			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t_{AD}	Address Delay										ns
	C = 90 pF			270			180			150	
	C = 30 pF			250			165			135	
t_{acc}	Peripheral Read Access Time $t_{acc} = t_{ut} - (t_{AD} + t_{DSR})$			530			360			250	ns
t_{DSR}	Data Set-up Time (Read)	100			60			40			ns
t_H	Input Data Hold Time	10			10			10			ns
t_H	Output Data Hold Time	10	25		10	25		10	25		ns
t_{AH}	Address Hold Time (Address, R/$\overline{W}$, VMA)	30	50		30	50		30	50		ns
t_{EH}	Enable HIGH Time for DBE Input	450			280			220			ns
t_{DDW}	Data Delay Time (Write)			225			200			160	ns
	Processor Controls										
t_{PCS}	Processor Control Set-up Time	200			140			110			ns
t_{PCr}, t_{PCf}	Processor Control Rise and Fall Time			100			100			100	ns
t_{BA}	Bus Available Delay			250			165			135	ns
t_{TSE}	3-State Enable			40			40			40	ns
t_{TSD}	3-State Delay			270			270			220	ns
$t_{\overline{DBE}}$	Data Bus Enable Down Time During $\phi1$ Up Time	150			120			75			ns
t_{DBEr}, t_{DBEf}	Data Bus Enable Rise and Fall Times			25			25			25	ns

Fig. 27 Clock Timing Waveform

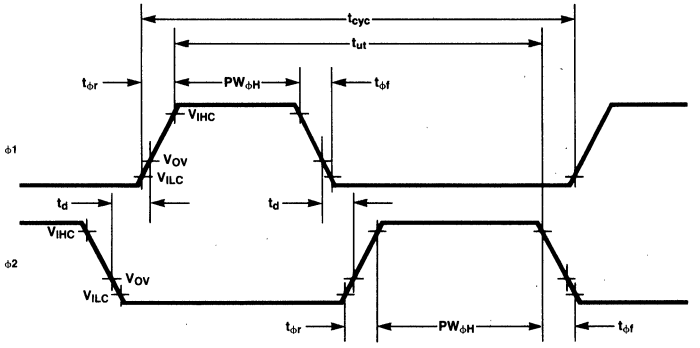


Fig. 28 Read Data From Memory or Peripherals

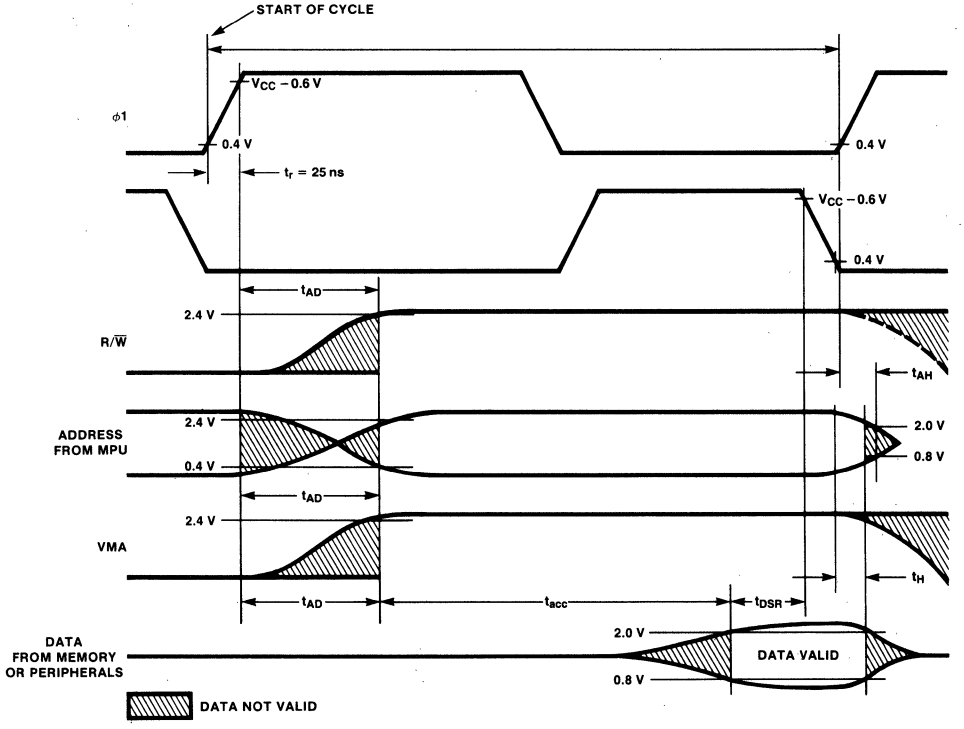


Fig. 29 Write In Memory or Peripherals

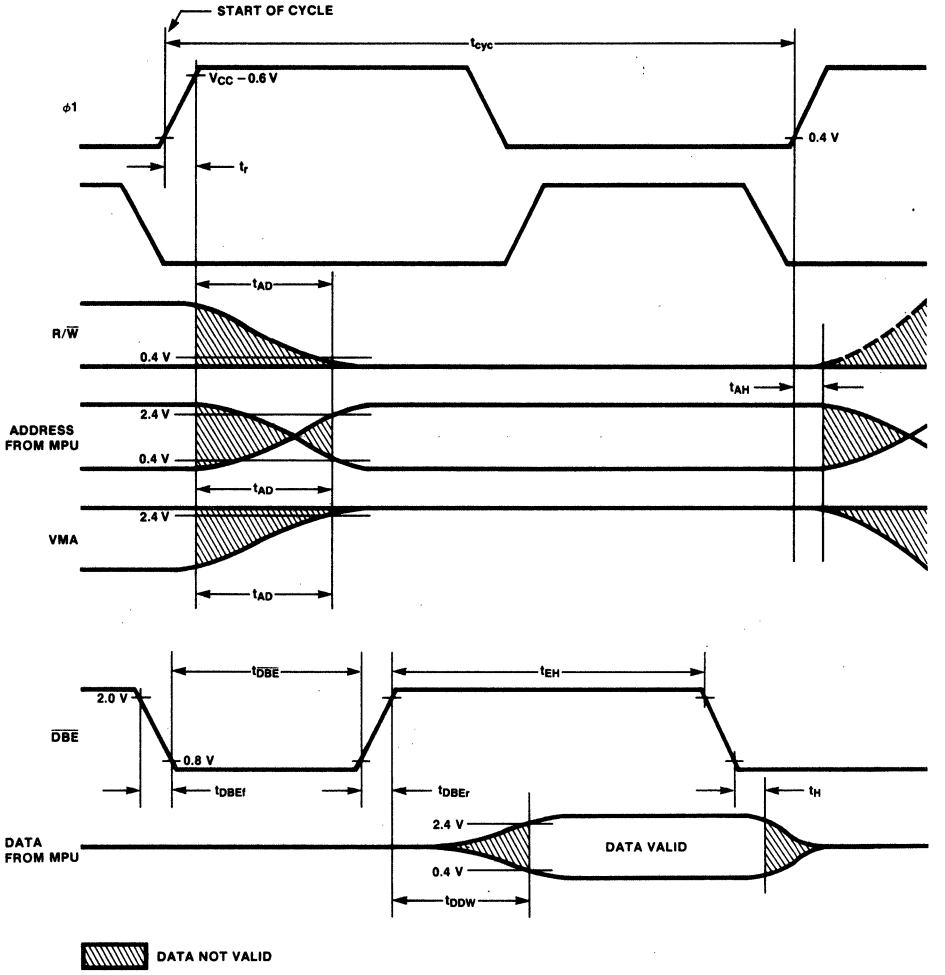


Fig. 30 Typical Data Bus Output Delay vs Capacitive Loading (t_{DDW})

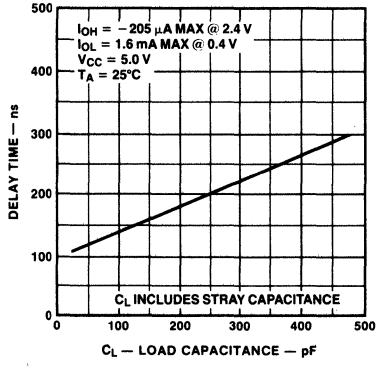


Fig. 31 Typical READ/WRITE, VMA, and Address Output Delay vs Capacitive Loading (t_{AD})

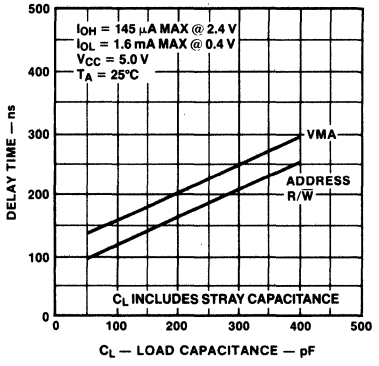
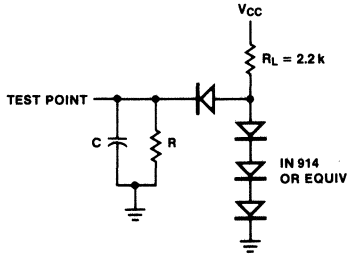


Fig. 32 Bus Timing Test Loads



Test Conditions

The dynamic test load for the data bus is 130 pF and one standard TTL load, as shown. The Address, R/$\overline{W}$, and VMA outputs are tested under two conditions to allow optimum operation in both buffered and unbuffered systems. The resistor (R) is chosen to insure specified load currents during V_{OH} measurement.

Notice that the data bus lines, the address lines, the interrupt request line, and the DBE line are all specified and tested to guarantee 0.4 V of dynamic noise immunity at both "1" and "0" logic levels.

C = 130 pF for D_0-D_7, E
= 90 pF for A_0-A_{15}, R/$\overline{W}$, and VMA (except t_{AD2})
= 30 pF for A_0-A_{15}, R/$\overline{W}$, and VMA (t_{AD2} only)
= 30 pF for BA.

R = 11.7 kΩ for D_0-D_7
= 16.5 kΩ for A_0-A_{15}, R/$\overline{W}$, and VMA
= 24 kΩ for BA

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6800P,S	0 to +70°C
	F6800CP,CS	−40 to +85°C
	F6800DM	−55 to +125°C
1.5 MHz	F68A00P,S	0 to +70°C
	F68A00C,CS	−40 to +85°C
	F68A00DM	−55 to +125°C
2.0 MHz	F68B00P,S	0 to +70°C
	F68B00C,CS	−40 to +85°C
	F68B00DM	−55 to +125°C

*P = plastic package, S = CER-DIP package.

5

FAIRCHILD
A Schlumberger Company

F6801/F6803
Single-Chip Microcomputer

Advance Product Information

Microprocessor Product

Description

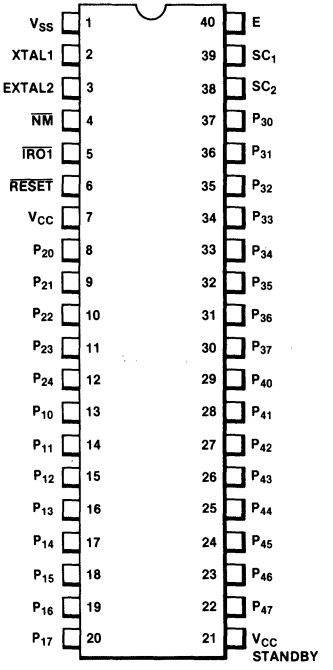
The Fairchild F6801/F6803 is an 8-bit single-chip microcomputer unit (MCU) which significantly enhances the capabilities of the F6800 family. It includes an upgraded F6800 microprocessor unit (MPU) with upward-source and object-code compatibility. The F6801/F6803 MCU can function as a monolithic microcomputer or can be expanded to a 64K byte address space. Features of the F6801/F6803 MCU include:

- **Enhanced F6800 Instruction Set (see table 1)**
- **8 × 8 Multiply Instruction**
- **Serial Communications Interface (SCI)**
- **16-bit Three-Function Programmable Timer**
- **Bus Compatibility with the F6800 Family**
- **2048 Bytes of ROM (F6801 Only)**
- **128 Bytes of RAM**
- **64 Bytes of RAM, Retainable During Powerdown**
- **29 Parallel I/O and Two Handshake Control Lines**
- **Internal Clock Generator With Divide-by-Four Output**
- **Interrupt Capability**
- **TTL compatible**
- **40-Pin Ceramic or Plastic Package**
- **+ 5V Power Supply**

The F6801/F6803 MCU can be configured to function in a wide variety of applications. This flexibility is provided by its ability to be hardware-programmed into eight different operating modes (see table 2). The operating mode controls the configuration of 18 of the 40 MCU pins, available on-chip resources, memory map, location of interrupt vectors, and type of external bus. Configuration of the remaining 22 pins is not dependent on the operating mode.

The F6803 can be considered an F6801 that operates in Modes 2 and 3 only (either internal RAM with no ROM, or no internal RAM or ROM, respectively).

Connection Diagram



5

Figure 1 Block Diagram

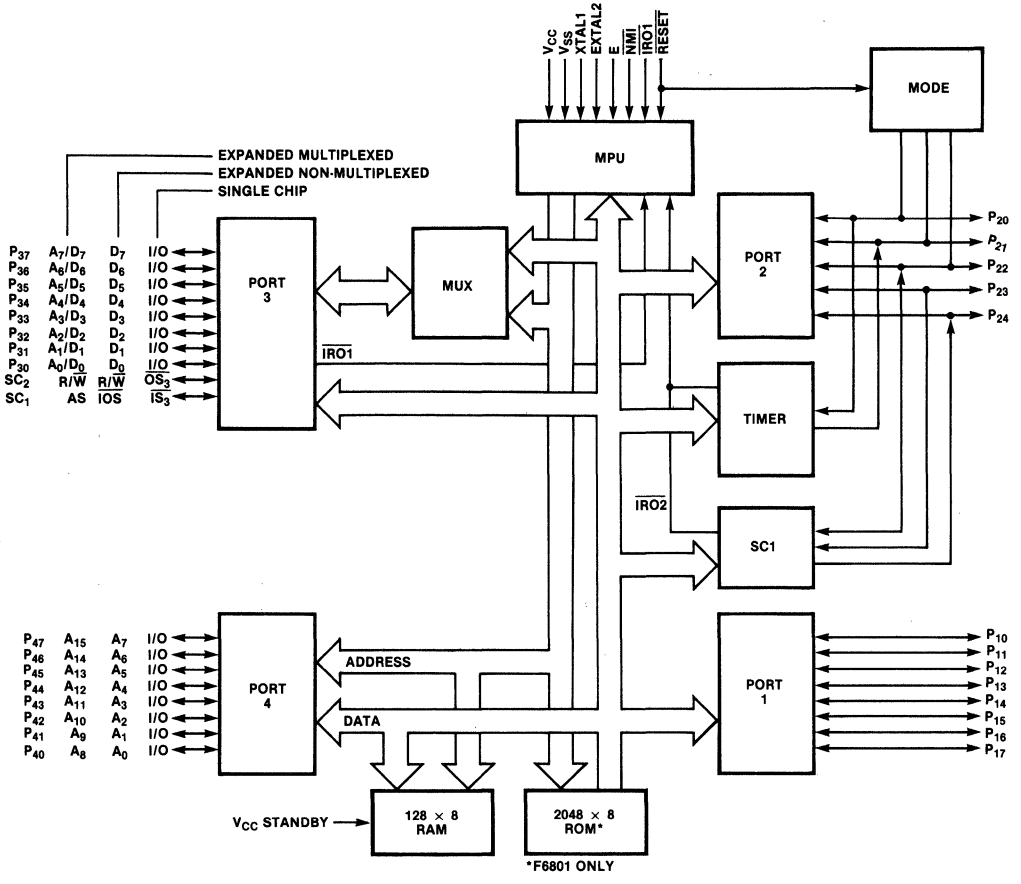
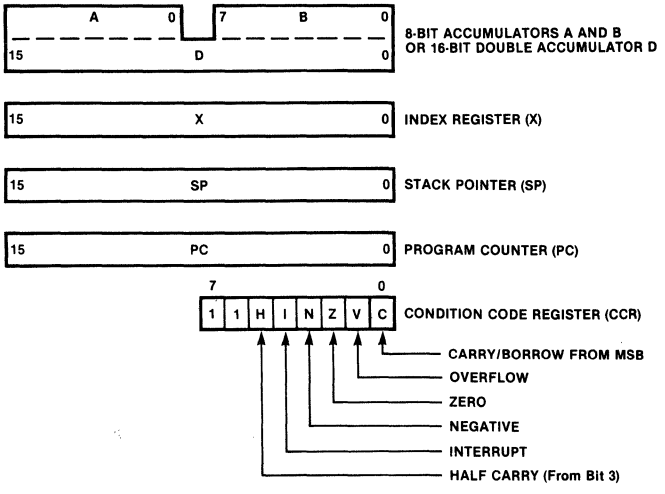


Figure 2 Programming Model



5

Table 1 New Instructions

Instruction	Description
ABX	Unsigned addition of Accumulator B to Index Register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD or LSLD	Shifts the double accumulator left (towards MSB) one bit, the LSB is cleared and the MSB is shifted into the C-bit
BHS	Branch if Higher or Same, unsigned conditional branch (same as BCC)
BLO	Branch if Lower. Unsigned conditional branch (same as BCS)
BRN	Branch Never
JSR	Additional addressing mode direct
LDD	Loads double accumulator from memory
LSL	Shifts memory or accumulator left(towards MSB) one bit, the LSB is cleared and the MSB ia shifted into the C-bit (same as ASL)
LSRD	Shifts the double accumulator right (towards LjSB) one bit, the MSB is cleared and the LSB is shifted into the C-bit
MUL	Unsigned multiply, multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the Index Register to stack
PULX	Pulls the Index Register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator
CPX	Internal processing modified to permit its use with any conditional branch instruction

Table 2 Summary of F6801/F6803 Operating Modes

Common to all Modes: Reserved Register Area Port 1 Port 2 Programmable Timer Serial Communications Interface
Single Chip Mode 7 128 bytes o RAMk, 2048 bytes of ROM Port 3 is a parallel I/O port with two control lines Port 4 is a parallel I/O port SC1 is Input Strobe 3 (IS3) SC2 is Output Strobe 3 (OS3)
Expanded Non-Multiplexed Mode 5 128 bytes of RAM, 2048 bytes of ROM 256 bytes of external memory space Port 3 is an 8-bit data bus Port 4 is an input port/address bus SC1 is Input/Output Select (IOS) SC2 is Read/Write (R/W)
Expanded Multiplexed Modes 1, 2, 3 6 Sour memory space options (64K address space) (1) No internal RAM or ROM (Mode 3) (2) Internal RAM, no ROM (Mode 2) (3) Internal RAM and ROM (Mode 1) (4) Internal RAM, ROM with partial address bus (Mode 6) Port 3 is a multiplexed address/data bus Port 4 is an address bus (inputs/address in Mode 6) SC1 is Address Strobe (AS) SC2 is Read/Write (R/W)
Test Modes 0 and 4 Expanded Multiplexed Test Mode 0 May be used to test RAM and ROM Single Chip and Non-Multiplexed Test Mode 4 (1) May be changed to Mode 5 without going through reset (2) May be used to test Ports 3 and 4 as I/O ports

5

Table 3 - Instruction Execution Times in E-Cycles

	Addressing Mode					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
ABA	•	•	•	•	2	•
ABX	•	•	•	•	3	•
ADC	2	3	4	4	•	•
ADD	2	3	4	4	•	•
ADDD	4	5	6	6	•	•
AND	2	3	4	4	•	•
ASL	•	•	6	6	2	•
ASLD	•	•	•	•	3	•
ASR	•	•	6	6	2	•
BCC	•	•	•	•	•	3
BCS	•	•	•	•	•	3
BEQ	•	•	•	•	•	3
BGE	•	•	•	•	•	3
BGT	•	•	•	•	•	3
BHI	•	•	•	•	•	3
BHS	•	•	•	•	•	3
BIT	2	3	4	4	•	•
BLE	•	•	•	•	•	3
BLO	•	•	•	•	•	3
BLS	•	•	•	•	•	3
BLT	•	•	•	•	•	3
BMI	•	•	•	•	•	3
BNE	•	•	•	•	•	3
BPL	•	•	•	•	•	3
BRA	•	•	•	•	•	3
BRN	•	•	•	•	•	3
BSR	•	•	•	•	•	6
BVC	•	•	•	•	•	3
BVS	•	•	•	•	•	3
CBA	•	•	•	•	2	•
CLC	•	•	•	•	2	•
CLI	•	•	•	•	2	•
CLR	•	•	6	6	2	•
CLV	•	•	•	•	2	•
CMP	2	3	4	4	•	•
COM	•	•	6	6	2	•
CPX	4	5	6	6	•	•
DAA	•	•	•	•	2	•
DEC	•	•	6	6	2	•
DES	•	•	•	•	3	•
DEX	•	•	•	•	3	•
EOR	2	3	4	4	•	•
INC	•	•	6	6	•	•
INS	•	•	•	•	3	•

	Addressing Mode					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
INX	•	•	•	•	3	•
JMP	•	•	3	3	•	•
JSR	•	5	6	6	•	•
LDA	2	3	4	4	•	•
LDD	3	4	5	5	•	•
LDS	3	4	5	5	•	•
LDX	3	4	5	5	•	•
LSL	•	•	6	6	2	•
LSLD	•	•	•	•	3	•
LSR	•	•	6	6	2	•
LSRD	•	•	•	•	3	•
MUL	•	•	•	•	10	•
NEG	•	•	6	6	2	•
NOP	•	•	•	•	2	•
ORA	2	3	4	4	•	•
PSH	•	•	•	•	3	•
PSHX	•	•	•	•	4	•
PUL	•	•	•	•	4	•
PULX	•	•	•	•	5	•
ROL	•	•	6	6	2	•
ROR	•	•	6	6	2	•
RTI	•	•	•	•	10	•
RTS	•	•	•	•	5	•
SBA	•	•	•	•	2	•
SBC	2	3	4	4	•	•
SEC	•	•	•	•	2	•
SEI	•	•	•	•	2	•
SEV	•	•	•	•	2	•
STA	•	3	4	4	•	•
STD	•	4	5	5	•	•
STS	•	4	5	5	•	•
STX	•	4	5	5	•	•
SUB	2	3	4	4	•	•
SUBD	4	5	6	6	•	•
SI	•	•	•	•	12	•
TAB	•	•	•	•	2	•
TAP	•	•	•	•	2	•
TBA	•	•	•	•	2	•
TPA	•	•	•	•	2	•
TST	•	•	6	6	2	•
TSX	•	•	•	•	3	•
TXS	•	•	•	•	3	•
WAI	•	•	•	•	9	•

FAIRCHILD
A Schlumberger Company

F6802/F6882/F6808 Microprocessor with Clock and RAM

Microprocessor Product

Description

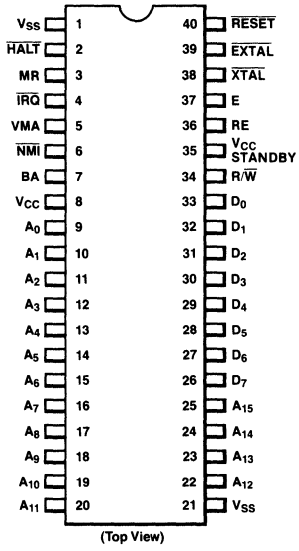
The F6802/F6882 is a monolithic 8-bit microprocessor that contains all the registers and accumulators of the F6800, plus an internal clock oscillator and driver on the same chip. The F6802/F6882 also has 128 bytes of RAM on board, located at hex addresses $0000 to $007F, V_{CC} standby can be utilized on the F6802/F6882 to facilitate memory retention during a power-down situation; the first 8 bytes of RAM at hex addresses $0000 to $0007 can be retained on the F6882, and the first 32 bytes of RAM at hex addresses $0000 to $001F can be retained on the F6802. The F6808 is identical to the F6802 without on-board RAM.

The F6802/F6882 is completely software-compatible with the F6800 microprocessor and the entire F6800 family of parts. (*Figure 1* illustrates a typical application using an F6800 family device.)

- **On-Chip Clock Circuit**
- **128 x 8-bit On-Chip RAM (Not Included on F6808)**
- **8 Bytes of RAM are Retainable on the F6882**
- **32 Bytes of RAM are Retainable on the F6802**
- **Software-Compatible with the F6800**
- **Standard TTL-Compatible Inputs and Outputs**
- **8-bit Bidirectional Data Bus**
- **16-bit Memory Addressing**
- **Interrupt Capability**
- **Speed Grades:**
  - **1.0 MHz F6802/F6882/F6808**
  - **1.5 MHz F68A02/F68A82/F68A08**

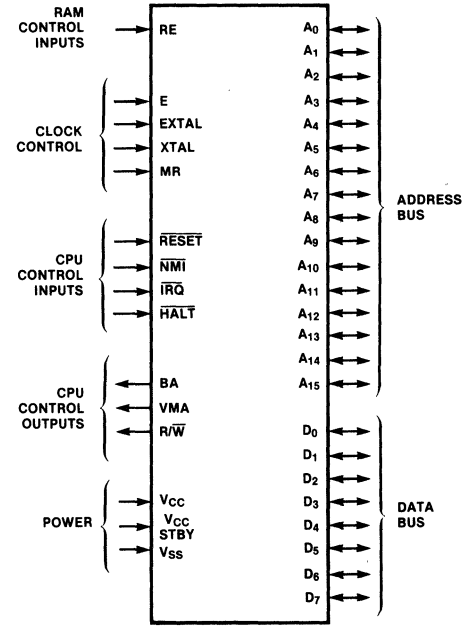
Connection Diagram

40-Pin DIP



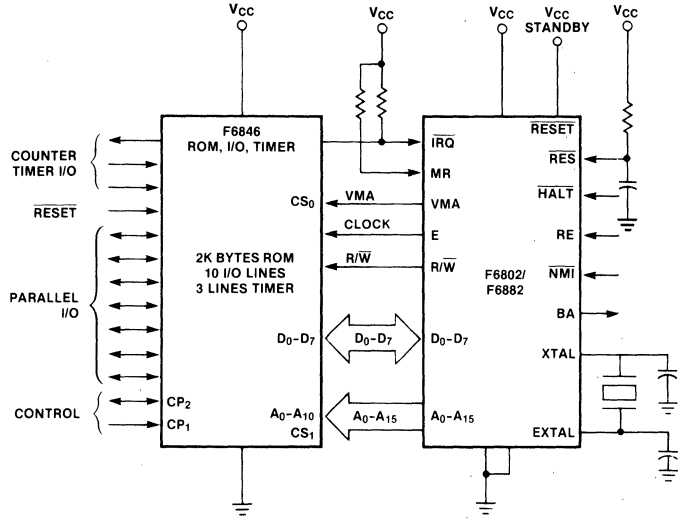
(Top View)

F6802/F6882/F6808 Signal Functions



5

Fig. 1 Typical Microcomputer Block Diagram



Registers

A general block diagram of the F6802/F6882 is shown in *Figure 2*. The number and configuration of the registers are identical to the F6800, as shown, with a 128 x 8-bit RAM* added to the basic microprocessor. The first 8 bytes in the F6882 and the first 32 bytes in the F6802 may be operated in a low-power mode via a V_{CC} standby and can be retained during power-up and power-down conditions via the RE signal. The F6808 is identical to the F6802 except for on-board RAM. Since the F6808 does not have on-board RAM, pin 36 must be tied to ground, allowing the processor to utilize up to 64K bytes of external memory.

The microprocessing unit (MPU) has three 16-bit registers and three 8-bit registers available for use by the programmer, as shown in *Figure 3.*

Program Counter

The program counter is a 2-byte (16-bit) register that points to the current program address.

Stack Pointer

The stack pointer is a 2-byte register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access read/write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be non-volatile.

Index Register

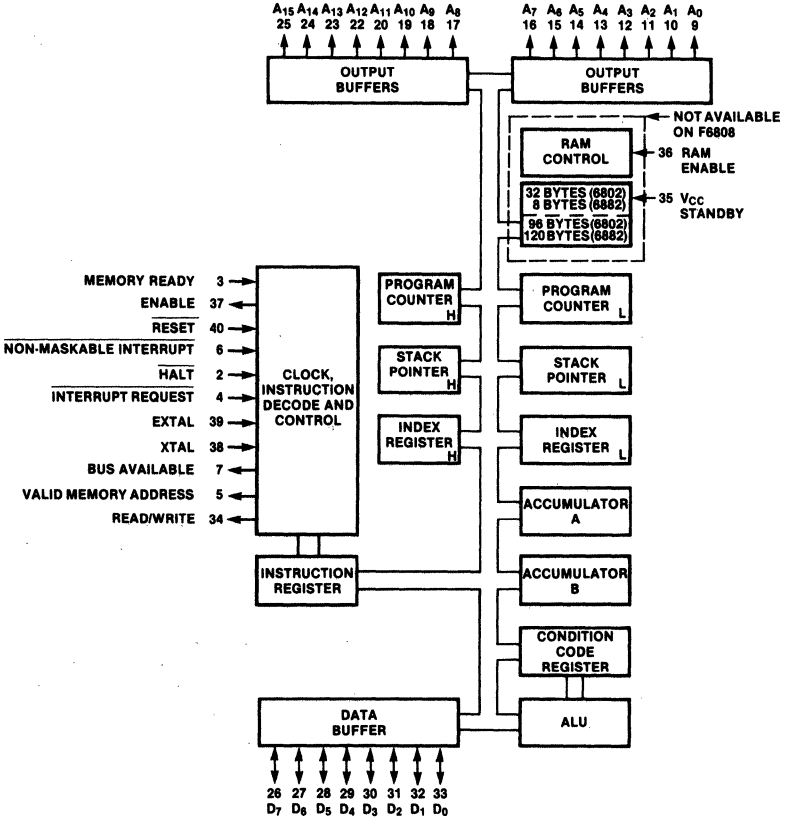
The index register is a 2-byte register that is used to store data or a 16-bit memory address for the indexed mode of memory addressing.

Accumulators

The two 8-bit accumulators are used to hold operands and results from an arithmetic logic unit (ALU).

*If programs are not executed from on-board RAM, TAV1 applies. If programs are to be stored and executed from on-board RAM, TAV2 applies. For normal data storage in the on-board RAM, this extended delay does not apply. Programs can be executed from on-board RAM when using 1.5 parts. On-board RAM can be used for data storage with all parts.

Fig. 2 F6802/F6882 Block Diagram



Condition Code Register (Status Word Register)
The condition code register indicates the results of an arithmetic logic unit operation: negative (N), zero (Z), overflow (V), carry from bit 7 (C), and half-carry from bit 3 (H). These bits of the condition code register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The unused bits of the condition code register (bit 6 and bit 7) are binary ones (1).

Figure 4 shows the order of saving the microprocessor status within the stack.

Fig. 3 Programming Model of the Microprocessing Unit

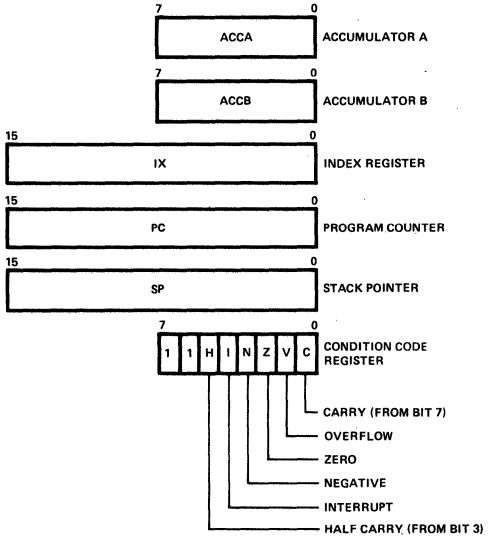
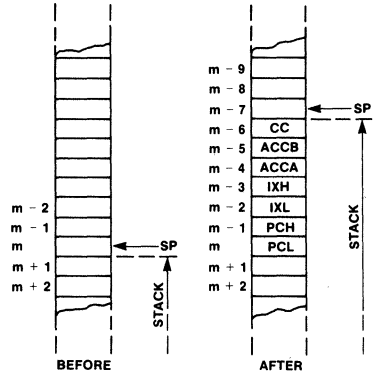


Fig. 4 Saving the Status of the Microprocessor in the Stack



Notes
SP = Stack Pointer
CC = Condition Code (also called the Processor Status Byte)
ACCB = Accumulator B
ACCA = Accumulator A
IXH = Index Register, higher order 8 bits
IXL = Index Register, lower order 8 bits
PCH = Program Counter, higher order 8 bits
PCL = Program Counter, lower order 8 bits

F6802/F6882 Signal Descriptions

The control and timing signals for the F6802/F6882 are identical to those of the F6800, with the following exceptions:

1. TSC, DBE ϕ_1, ϕ_2 input, and two unused pins have been eliminated.
2. The following signal and timing lines have been added:
   RAM Enable (RE)
   Crystal Connections EXtal and Xtal
   Memory Ready (MR)
   V_{CC} Standby
   Enable ϕ_2 Output (E)

The following summarizes the F6802/F6882 MPU signals.

Data Bus

D_0-D_7 (Data Bus Lines), Pins 26-33
Bidirectional bus used to transfer data to and from the memory and peripheral devices. Also has 3-state output buffers capable of driving one standard TTL load and 130 pF.

Data bus lines are in the output mode when the internal RAM is accessed. This prohibits external data from entering the MPU. The internal RAM is fully decoded from addresses $0000 to $007F. External RAM at $0000 to $007F must be disabled when internal RAM is accessed.

Address Bus

A_0-A_{15} (Address Bus Lines), Pins 9-20, 22-25
Sixteen output lines form the address bus. The outputs are capable of driving one standard TTL load and 90 pF. These lines do not have 3-state capability.

CPU Control Inputs

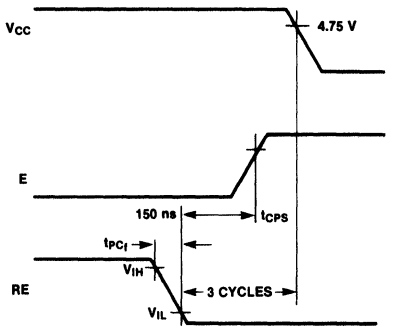
$\overline{\text{RESET}}$ (Reset), Pin 40
Input used to reset and start the MPU from a power-down condition resulting from a power failure or an initial start-up of the processor. When this line is low,the MPU is inactive and the information in the registers is lost. If a high level is detected on the input, this signals the MPU to begin the restart sequence. This starts execution of a routine to initialize the processor from its reset condition. All the higher order address lines are forced high. For the restart, the last two locations in memory

($FFFE, $FFFF) are used to load the program that is addressed by the program counter. During the restart routine, the interrupt mask bit is set and must be reset before the MPU can be interrupted by $\overline{IRQ}$. Power-up and reset timing sequences are shown in *Figures 5* and *6*.

When brought low, $\overline{RESET}$ *must be held low at least three clock cycles.* This is independent of the power-up delay required for oscillator start-up (T_{RC}).

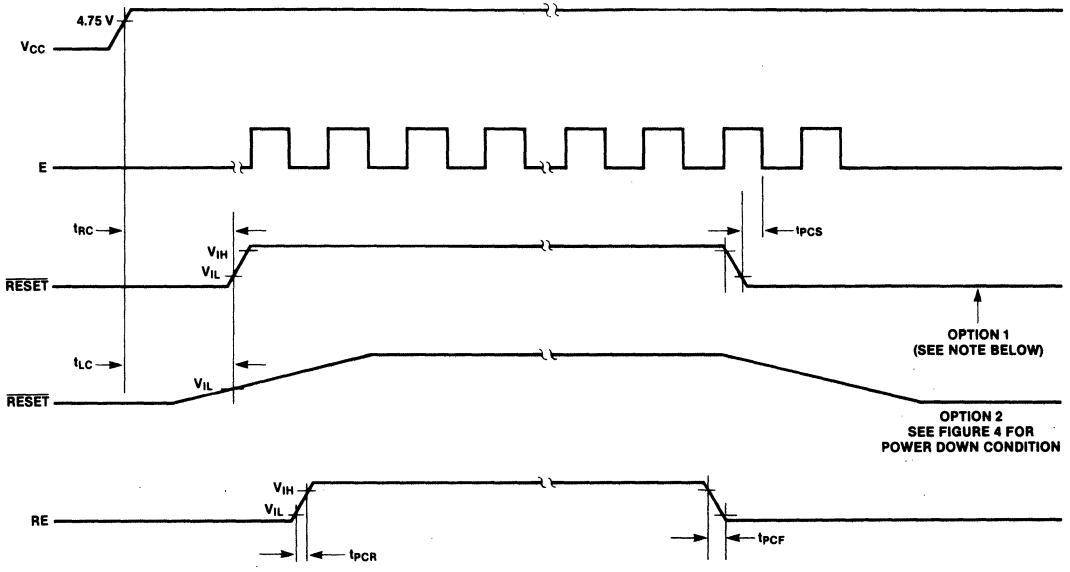
When $\overline{RESET}$ is released, it must go through the low-to-high threshold without bouncing, oscillating, or otherwise causing an erroneous reset (less than three clock cycles) that may cause improper MPU operation until the next valid reset.

Fig. 6 Power-Down Sequence



5

Fig. 5 Power-Up and Reset Timing



Note
If option 1 is chosen, $\overline{RESET}$ and RE pins can be tied together.

$\overline{NMI}$ (Non-Maskable Interrupt), Pin 6
A low-going edge on this input requests that a non-maskable interrupt sequence be generated within the processor. As with the interrupt request ($\overline{IRQ}$) signal, the processor completes the current instruction being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the condition code register has no effect on $\overline{NMI}$.

The index register, program counter, accumulators, and condition code register are stored on the stack, as shown in *Figure 4*. At the end of the cycle, a 16-bit address will be loaded from memory locations $FFFC and $FFFD that points to a vectoring address. An address loaded from these locations causes the MPU to branch to a non-maskable interrupt routine in memory.

A nominal 3 kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts. The $\overline{NMI}$ signal may be tied directly to V_{CC} if not used.

The $\overline{IRQ}$ and $\overline{NMI}$ inputs are hardware interrupt lines that are sampled when E is high and start the interrupt routine on a low E following the completion of an instruction.

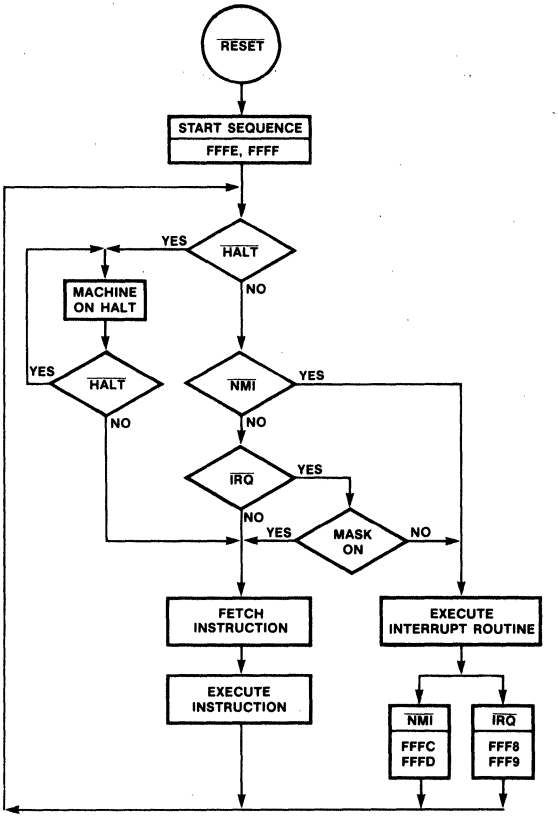
Figure 7 is a flow chart describing the major decision paths and interrupt vectors of the microprocessor. *Table 1* gives the memory map for interrupt vectors.

Table 1 Memory Map for Interrupt Vectors

Vector		
MS	LS	Description
$FFFE	$FFFF	Restart
$FFFC	$FFFD	Non-Maskable Interrupt
$FFFA	$FFFB	Software Interrupt
$FFF8	$FFF9	Interrupt Request

$\overline{IRQ}$ (Interrupt Request), Pin 4
This level-sensitive input requests that an interrupt sequence be generated within the machine. The processor waits until it completes the current instruction that is being executed before it recognizes the request. At that time, if the interrupt mask bit in the condition code register is not set, the machine begins an interrupt sequence. The index register, program counter, accumulators, and condition code register are stored on the stack as shown in *Figure 4*. The MPU responds to the interrupt request by setting the interrupt mask bit high

Fig. 7 MPU Flow Chart



so that no further interrupts may occur. At the end of the cycle, a 16-bit address is loaded from memory locations $FFF8 and $FFF9 that point to a vectoring address. An address loaded from these locations causes the MPU to branch to an interrupt routine in memory.

The $\overline{HALT}$ line must be in the high state for interrupts to be serviced. Interrupts are latched internally while $\overline{HALT}$ is low.

The $\overline{IRQ}$ has a high-impedance pull-up device internal to the chip; however, a 3kΩ external resistor to V_{CC} should be used for the wire-OR and optimum control of interrupts.

$\overline{\text{HALT}}$ (Halt), Pin 2
When this input is in the low state, all activity in the machine is halted. This input is level-sensitive. In the halt mode, the machine stops at the end of an instruction. Bus Available is in a high state, and Valid Memory Address is in a low state. The address bus displays the address of the next instruction.

To ensure single-instruction operation, transition of the $\overline{\text{HALT}}$ line must occur t_{PCS} before the falling edge of E and the $\overline{\text{HALT}}$ line must go high for one clock cycle.

$\overline{\text{HALT}}$ should be tied high if not used.

RAM Control Port

RE (RAM Enable), Pin 36
A TTL-compatible RAM enable input that controls the on-chip RAM. When placed in the high state, the on-chip memory is enabled to respond to the MPU controls. In the low state, the RAM is disabled. This pin may also be utilized to disable reading from and writing to the on-chip RAM during a power-down situation. The RE signal must be low three cycles before V_{CC} goes below 4.75 V during power-down as shown in *Figure 6.*

The RE signal should be tied low on the F6808; it should be tied to the correct high or low state if not used.

CPU Control Outputs

BA (Bus Available), Pin 7
Is normally in the low state; when activated, it goes to the high state, indicating that the microprocessor has stopped and that the address bus is available (but not in a 3-state condition). This occurs if the $\overline{\text{HALT}}$ line is in the low state or the processor is in the wait state as a result of execution of a WAIT instruction. At such time, all 3-state output drivers go to their off state and other outputs to their normally inactive levels. The processor is removed from the wait state by the occurrence of a maskable (mask bit 1 = 0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30 pF.

VMA (Valid Memory Address), Pin 5
This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces, such as the PIA and ACIA. This signal is not 3-state. One standard TTL load and 90 pF may be directly driven by this active-high signal.

R/$\overline{\text{W}}$ (Read, Write), Pin 34
This TTL-compatible output signals the periphals and memory devices whether the MPU is in a read (high) or write (low) state. The normal standby state of this signal is read (high). When the processor is halted, it is in the read state. This output is capable of driving one standard TTL load and 90 pF.

Power

V_{CC} (Power Supply), Pin 8
V_{CC} tolerance is ±5%.

V_{CC} STBY (Power Supply Standby), Pin 35
This pin supplies the dc voltage to the first 8 or 32 bytes of RAM as well as the RAM enable (RE) control logic. Thus, retention of data in this portion of the RAM on a power-up, power-down, or standby condition is guaranteed. Maximum current drain at maximum V_{SB} is I_{SBB}.

V_{SS} (Ground), Pins 1, 21
System ground; 0 V reference.

Clock Control

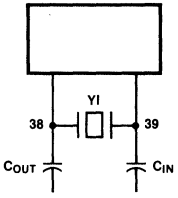
E (Enable), Pin 37
This pin supplies the clock for the MPU and the rest of the system. This is a single-phase, TTL-compatible clock and may be conditioned by a memory ready (MR) signal. The E signal is equivalent to ϕ_2 on the F6800, and is capable of driving one TTL load and 130 pF.

EXTAL (External Crystal Connector), Pin 39
XTAL (Crystal Connector), Pin 38
The F6802/F6882 has an internal oscillator that may be crystal controlled. These connections are for a parallel-resonant, AT cut, fundamental crystal. (*Figure 8* illustrates the crystal specifications.) A divide-by-four circuit has been added so that a 4 MHz crystal may be used in place of a 1 MHz crystal for a more cost-effective system. An example of the crystal circuit layout on a printed circuit board is shown in *Figure 9.*

Pin 39 may be driven externally by a TTL-input signal four times the required clock frequency. Pin 38 is to be grounded in this mode.

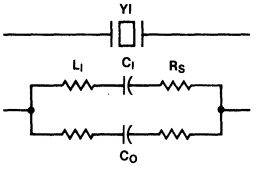
An RC network is not directly usable as a frequency source on pins 38 and 39. An RC network-type TTL or CMOS oscillator works well as long as the TTL or CMOS output drives the on-chip oscillator.

Fig. 8 Crystal Specification



YI	C_{IN}	C_{OUT}
3.58 MHz	27 pF	27 pF
4 MHz	27 pF	27 pF
6 MHz	20 pF	20 pF
8 MHz	18 pF	18 pF

Crystal Loading

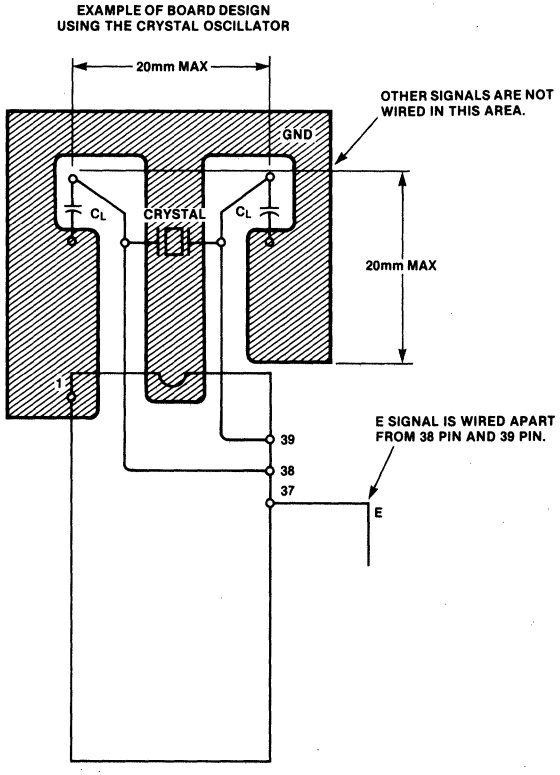


	3.58 MHz	4.0 MHz	6.0 MHz	8.0 MHz
R_S	60Ω	50Ω	30-50Ω	20-40Ω
C_O	3.5 pF	6.5 pF	4-6 pF	4-6 pF
C_I	0.015 pF	0.025 pF	0.01-0.02 pF	0.01-0.02 pF
Q	>40K	>30K	>20K	>20K

Nominal Crystal Parameters*

*These are representative AT-cut parallel-response crystal parameters only. Crystals of other types of cuts may also be used.

Fig. 9 Suggested PC Board Layout



LC networks in place of the crystal are not recommended.

If an external clock is used, it may be halted for more than $PWOL. The F6802/F6882/F6808 is a dynamic part except for the internal RAM, and requires the external clock to retain information.

MR (Memory Ready), Pin 3

A TTL-compatible input control signal that allows stretching of the enable (E) signal. Use of MR requires synchronization with the $4xf_o$ signal, as shown in *Figure 10.* When MR is high, E will be in normal operation. When MR is low, E may be stretched integral multiples of half periods, allowing interface to slow memories or peripherals. A maximum stretch is t_{CYC}; The MR signal should be tied high if not used. Refer to *Figure 11* for MR timing information.

Fig. 10 Memory Ready Synchronization

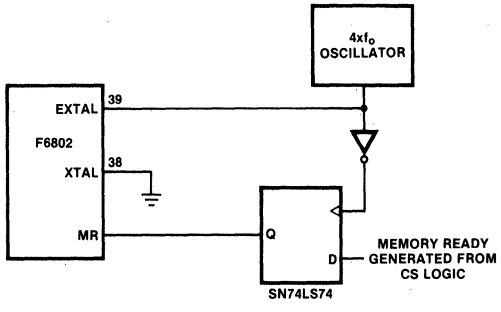
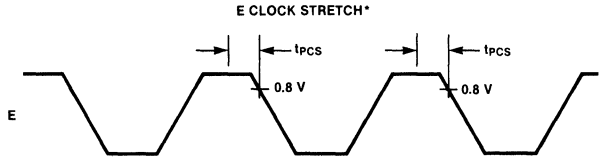
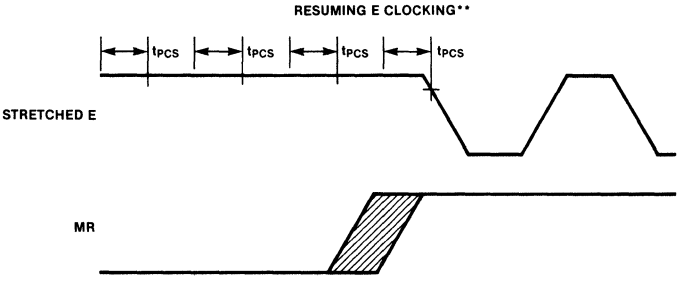


Fig. 11 MR Negative Setup Time Requirement



*The E clock will be stretched at the end of E high of the cycle during which MR negative meets the t_{PCS} setup time. The t_{PCS} setup time is referenced to the fall of E. If the t_{PCS} setup time is not met, E is stretched at the end of the next E high one-half cycle. The E signal isstretched in integral multiples of one-half cycles.



**The E clock resumes normal operation at the end of the one-half cycle during which MR assertion meets the t_{PCS} setup time. The t_{PCS} setup time is referenced to transitions of E were it not stretched. If t_{PCS} setup time is not met, E falls at the second possible transition time after MR is asserted. There is no direct means of determining when the t_{PCS} references occur, unless the synchronizing circuit of *Figure 10* is used.

5

MPU Instruction Set

The F6802/F6882 has a basic set of 70 instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt, and stack manipulation instructions (refer to *Tables 2* through *6*); special operations are illustrated in *Figure 12*. This instruction set is identical to that of the F6800.

MPU Addressing Modes

There are seven address modes that can be used by a programmer. The addressing modes can be used as a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in *Table 7*, along with the associated instruction execution time that is given in machine cycles. With a bus frequency of 1 MHz, these times would be microseconds.

Accumulator (ACCX) Addressing

In accumulator-only addressing, either accumulator A or accumulator B is specified. These are 1-byte instructions.

Immediate Addressing

In immediate addressing, the operand is contained in the second byte of the instruction. The exceptions are LDS and LDX, which have the operand in the second and third bytes of the instruction. The MPU addresses this location when it fetches the immediate instruction for execution. These are 2- or 3-byte instructions.

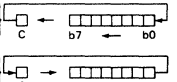
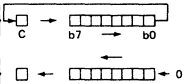
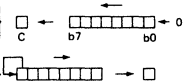
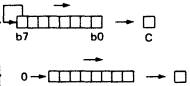
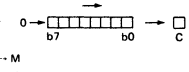
Direct Addressing

In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine (i.e., locations zero through 255). Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random access memory. These are 2-byte instructions.

Table 2 F6802/6882 Microprocessor Instruction Set—Alphabetic Sequence

ABA	Add Accumulators
ADC	Add with Carry
ADD	Add
AND	Logical And
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
BCC	Branch if Carry Clear
BCS	Branch if Carry Set
BEQ	Branch if Equal to Zero
BGE	Branch if Greater or Equal Zero
BGT	Branch if Greater than Zero
BHI	Branch if Higher
BIT	Bit Test
BLE	Branch if Less or Equal
BLS	Branch if Lower or Same
BLT	Branch if Less than Zero
BMI	Branch if Minus
BNE	Branch if Not Equal to Zero
BPL	Branch if Plus
BRA	Branch Always
BSR	Branch to Subroutine
BVC	Branch if Overflow Clear
BVS	Branch if Overflow Set
CBA	Compare Accumulators
CLC	Clear Carry
CLI	Clear Interrupt Mask
CLR	Clear
CLV	Clear Overflow
CMP	Compare
COM	Complement
CPX	Compare Index Register
DAA	Decimal Adjust
DEC	Decrement
DES	Decrement Stack Pointer
DEX	Decrement Index Register
EOR	Exclusive OR
INC	Increment
INS	Increment Stack Pointer
INX	Increment Index Register
JMP	Jump
JSR	Jump to Subroutine
LDA	Load Accumulator
LDS	Load Stack Pointer
LDX	Load Index Register
LSR	Logical Shift Right
NEG	Negate
NOP	No Operation
ORA	Inclusive OR Accumulator
PSH	Push Data
PUL	Pull Data
ROL	Rotate Left
ROR	Rotate Right
RTI	Return form Interrupt
RTS	Return from Subroutine
SBA	Subtract Accumulators
SBC	Subtract with Carry
SEC	Set Carry
SEI	Set Interrupt Mask
SEV	Set Overflow
STA	Store Accumulator
STS	Store Stack Register
STX	Store Index Register
SUB	Subtract
SWI	Software Interrupt
TAB	Transfer Accumulators
TAP	Transfer Accumulators to Condition Code Reg.
TBA	Transfer Accumulators
TPA	Transfer Condition Code Reg. to Accumulator
TST	Test
TSX	Transfer Stack Pointer to Index Register
TXS	Transfer Index Register to Stack Pointer
WAI	Wait for Interrupt

Table 3 Accumulator and Memory Instructions

OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED =	DIRECT OP	DIRECT ~	DIRECT =	INDEX OP	INDEX ~	INDEX =	EXTND OP	EXTND ~	EXTND =	IMPLIED OP	IMPLIED ~	IMPLIED =	BOOLEAN/ARITHMETIC OPERATION (All register labels refer to contents)	5 H	4 I	3 N	2 Z	1 V	0 C
Add	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B · M	•	•	↕	↕	R	•
Clear	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
Compare Acmltrs	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Complement, 1s	COM							63	7	2	73	6	3				$\overline{M}$ → M	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A}$ → A	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B}$ → B	•	•	↕	↕	R	S
Complement, 2s (Negate)	NEG							60	7	2	70	6	3				00 − M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts Binary Add. of BCD Characters into BCD Format	•	•	↕	↕	↕	③
Decrement	DEC							6A	7	2	7A	6	3				M − 1 → M	•	•	↕	↕	4	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	4	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	4	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	↕	↕	R	•
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	4	1	A → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, M_{SP} → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, M_{SP} → B	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	ROLA													49	2	1		•	•	↕	↕	⑥	↕
	ROLB													59	2	1		•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	7	2	76	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	RORA													46	2	1		•	•	↕	↕	⑥	↕
	RORB													56	2	1		•	•	↕	↕	⑥	↕
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	ASLA													48	2	1		•	•	↕	↕	⑥	↕
	ASLB													58	2	1		•	•	↕	↕	⑥	↕
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	ASRA													47	2	1		•	•	↕	↕	⑥	↕
	ASRB													57	2	1		•	•	↕	↕	⑥	↕
Shift Right, Logic	LSR							64	7	2	74	6	3				M, A, B 	•	•	R	↕	⑥	↕
	LSRA													44	2	1		•	•	R	↕	⑥	↕
	LSRB													54	2	1		•	•	R	↕	⑥	↕
Store Acmltr	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A − M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B − M → B	•	•	↕	↕	↕	↕
Subtract Acmltrs	SBA													10	2	1	A − B → A	•	•	↕	↕	↕	↕
Subtr. with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A − M − C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B − M − C → B	•	•	↕	↕	↕	↕
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	7	2	7D	6	3				M − 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A − 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B − 00	•	•	↕	↕	R	R
																		H	I	N	Z	V	C

Note
Accumulator addressing mode instructions are included in the column for implied addressing.

Legend

OP	Operation code (hexadecimal)
~	Number of MPU cycles
=	Number of program bytes
+	Arithmetic plus
−	Arithmetic minus
*	Boolean AND
M_{SP}	Contents of memory location point to be stack pointer
+	Boolean inclusive-OR
−	Boolean exclusive-OR
M	Complement of M
→	Transfer into
0	Bit = zero
00	Byte = zero

Condition Code Symbols

H	Half-carry from bit 3
I	Interrupt mask
N	Negative (sign bit)
Z	Zero (byte)
V	Overflow, 2's complement
C	Carry from bit 7
R	Reset always
S	Set always
:	Test and set if true, cleared otherwise
•	Not affected

Table 4 Index Register and Stack Manipulation Instructions

		IMMED			DIRECT			INDEX			EXTND			IMPLIED				COND. CODE REG.					
																		5	4	3	2	1	0
POINTER OPERATIONS	MNEMONIC	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	BOOLEAN/ARITHMETIC OPERATION	H	I	N	Z	V	C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M+1)$	•	•	⑦	↕	⑧	•
Decrement Index Reg	DEX													09	4	1	X - 1 → X	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	SP - 1 → SP	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	SP + 1 → SP	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	⑨	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	⑨	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Indx Reg → Stack Pntr	TXS													35	4	1	X - 1 → SP	•	•	•	•	•	•
Stack Pntr → Indx Reg	TSX													30	4	1	SP + 1 → X	•	•	•	•	•	•

Table 5 Jump and Branch Instructions

		RELATIVE			INDEX			EXTND			IMPLIED				COND. CODE REG.						
															5	4	3	2	1	0	
OPERATIONS	MNEMONIC	OP	~	#	OP	~	#	OP	~	#	OP	~	#	BRANCH TEST	H	I	N	Z	V	C	
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•	
Branch If Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•	
Branch If Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•	
Branch If = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•	
Branch If ≥ Zero	BGE	2C	4	2										N⊕V = 0	•	•	•	•	•	•	
Branch If > Zero	BGT	2E	4	2										Z + (N⊕V) = 0	•	•	•	•	•	•	
Branch If Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•	
Branch If ≤ Zero	BLE	2F	4	2										Z + (N⊕V) = 1	•	•	•	•	•	•	
Branch If Lower Or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•	
Branch If < Zero	BLT	2D	4	2										N⊕V = 1	•	•	•	•	•	•	
Branch If Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•	
Branch If Not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•	
Branch If Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•	
Branch If Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•	
Branch If Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•	
Branch To Subroutine	BSR	8D	8	2											•	•	•	•	•	•	
Jump	JMP				6E	4	2	7E	3	3				See Special Operations	•	•	•	•	•	•	
Jump To Subroutine	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•	
No Operation	NOP										01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•	
Return From Interrupt	RTI										3B	10	1		——			⑩		——	
Return From Subroutine	RTS										39	5	1		•	•	•	•	•	•	
Software Interrupt	SWI										3F	12	1	See Special Operations	•	•	•	•	•	•	
Wait for Interrupt*	WAI										3E	9	1		•	⑪	•	•	•	•	

*WAI puts address bus, R/$\overline{W}$, and data bus in the 3-state mode while VMA is held low.

Table 6 Condition Code Register Manipulation Instructions

OPERATIONS	MNEMONIC	IMPLIED OP	~	#	BOOLEAN OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Acmltr A → CCR	TAP	06	2	1	A → CCR	———			(12)		———
CCR → Acmltr A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

Condition Code Register Notes (Bit set if test is true and cleared otherwise)

1 (Bit V) Test: Result = 10000000?
2 (Bit C) Test: Result = 00000000?
3 (Bit C) Test: Decimal value of most significant BCD character greater than nine? (Not cleared if previously set.)
4 (Bit V) Test: Operand = 10000000 prior to execution?
5 (Bit V) Test: Operand = 01111111 prior to execution?
6 (Bit V) Test: Set equal to result of N ⊕ C after shift has occurred.
7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?
8 (Bit V) Test: 2's complement overflow from subtraction of MS bytes?
9 (Bit N) Test: Result less than zero? (Bit 15 = 1)
10 (All) Load condition code register from stack (see Figure 10)
11 (Bit I) Set when interrupt occurs. If previously set, a non-maskable interrupt is required to exit the wait state.
12 (All) Set according to the contents of accumulator A.

5

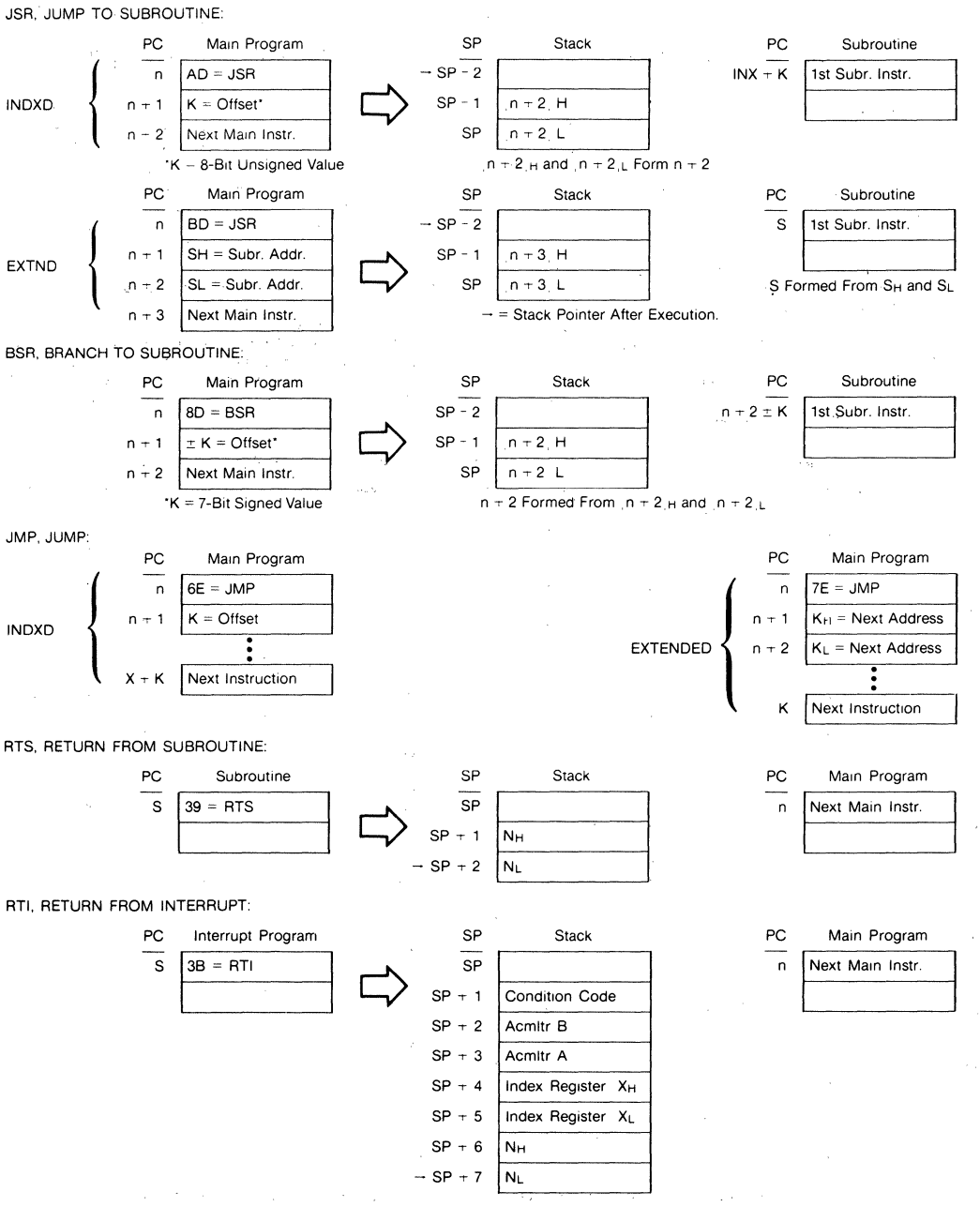
Table 7 Instruction Addressing Modes and Associated Execution Times (Times in Machine Cycles)

	(Dual Operand)	ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative
ABA		•	•	•	•	•	2	•
ADC	x	•	2	3	4	5	•	•
ADD	x	•	2	3	4	5	•	•
AND	x	•	2	3	4	5	•	•
ASL		2	•	•	6	7	•	•
ASR		2	•	•	6	7	•	•
BCC		•	•	•	•	•	•	4
BCS		•	•	•	•	•	•	4
BEA		•	•	•	•	•	•	4
BGE		•	•	•	•	•	•	4
BGT		•	•	•	•	•	•	4
BHI		•	•	•	•	•	•	4
BIT	x	•	2	3	4	5	•	•
BLE		•	•	•	•	•	•	4
BLS		•	•	•	•	•	•	4
BLT		•	•	•	•	•	•	4
BMI		•	•	•	•	•	•	4
BNE		•	•	•	•	•	•	4
BPL		•	•	•	•	•	•	4
BRA		•	•	•	•	•	•	4
BSR		•	•	•	•	•	•	8
BVC		•	•	•	•	•	•	4
BVS		•	•	•	•	•	•	4
CBA		•	•	•	•	•	2	•
CLC		•	•	•	•	•	2	•
CLI		•	•	•	•	•	2	•
CLR		2	•	•	6	7	•	•
CLV		•	•	•	•	•	2	•
CMP	x	•	2	3	4	5	•	•
COM		2	•	•	6	7	•	•
CPX		•	3	4	5	6	•	•
DAA		•	•	•	•	•	2	•
DEC		2	•	•	6	7	•	•
DES		•	•	•	•	•	4	•
DEX		•	•	•	•	•	4	•
EOR	x	•	2	3	4	5	•	•

	(Dual Operand)	ACCX	Immediate	Direct	Extended	Indexed	Implied
INC		2	•	•	6	7	•
INS		•	•	•	•	•	4
INX		•	•	•	•	•	4
JMP		•	•	•	3	4	•
JSR		•	•	•	9	8	•
LDA	x	•	2	3	4	5	•
LDS		•	3	4	5	6	•
LDX		•	3	4	5	6	•
LSR		2	•	•	6	7	•
NEG		2	•	•	6	7	•
NOP		•	•	•	•	•	2
ORA	x	•	2	3	4	5	•
PSH		•	•	•	•	•	4
PUL		•	•	•	•	•	4
ROL		2	•	•	6	7	•
ROR		2	•	•	6	7	•
RTI		•	•	•	•	•	10
RTS		•	•	•	•	•	5
SBA		•	•	•	•	•	2
SBC	x	•	2	3	4	5	•
SEC		•	•	•	•	•	2
SEI		•	•	•	•	•	2
SEV		•	•	•	•	•	2
STA	x	•	•	4	5	6	•
STS		•	•	5	6	7	•
STX		•	•	5	6	7	•
SUB	x	•	2	3	4	5	•
SWI		•	•	•	•	•	12
TAB		•	•	•	•	•	2
TAP		•	•	•	•	•	2
TBA		•	•	•	•	•	2
TPA		•	•	•	•	•	2
TST		2	•	•	6	7	•
TSX		•	•	•	•	•	4
TSX		•	•	•	•	•	4
WAI		•	•	•	•	•	9

Note
Interrupt time is 12 cycles from the end of the instruction being executed, except following a WAI instruction, when it is four cycles.

Fig. 12 Special Operations



Extended Addressing
In extended addressing, the address contained in the second byte of the instruction is used as the higher 8 bits of the address of the operand. The third byte of the instruction is used as the lower 8 bits of the address for the operand. This is an absolute address in memory. These are 3-byte instructions.

Indexed Addressing
In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest 8 bits in the MPU. The carry is then added to the higher order 8 bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are 2-byte instructions.

Implied Addressing
In the implied addressing mode, the instructions give the address (i.e., stack pointer, index register, etc.). These are 1-byte instructions.

Relative Addressing
In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest 8 bits plus two. The carry or borrow is then added to the higher 8 bits. This allows the user to address data within a range of −125 to +129 bytes of the present instruction. These are 2-byte instructions.

Summary of Cycle-by-Cycle Operation
Table 8 provides a detailed description of the information present on the address bus, data bus, valid memory address (VMA) line, and the read/write (R/$\overline{W}$) line during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized according to addressing mode and number of cycles per instruction. (In general, instructions with the same addressing mode and number of cycles execute in the same manner. Exceptions are indicated in the table.)

Output Delay

Figures 13 and *14* illustrate typical output delays versus capacitance loading.

Fig. 13 Typical Data Bus Output Delay vs. Capacitive Loading

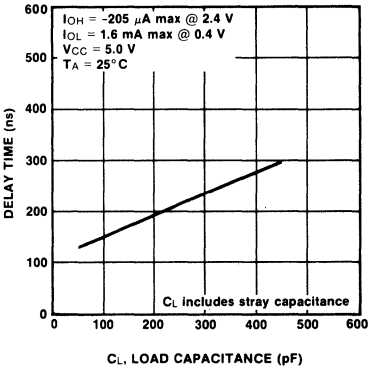


Fig. 14 Typical Read/$\overline{\text{Write}}$, VMA and Address Output Delay vs. Capacitive Loading

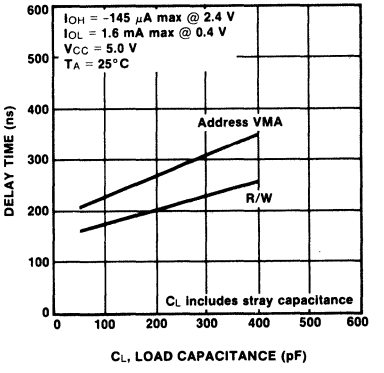


Table 8 Operation Summary

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/W Line	Data Bus
Immediate						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	1 1	Op Code Address Op Code Address + 1	1 1	Op Code Operand Data
CPX LDS LDX	3	1 2 3	1 1 1	Op Code Address Op Code Address + 1 Op Code Address + 2	1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte)
Direct						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	1 1 1	Op Code Address Op Code Address + 1 Address of Operand	1 1 1	Op Code Address of Operand Operand Data
CPX LDS LDX	4	1 2 3 4	1 1 1 1	Op Code Address Op Code Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STA	4	1 2 3 4	1 1 0 1	Op Code Address Op Code Address + 1 Destination Address Destination Address	1 1 1 0	Op Code Destination Address Irrelevant Data (Note 1) Data from Accumulator
STS STX	5	1 2 3 4 5	1 1 0 1 1	Op Code Address Op Code Address + 1 Address of Operand Address of Operand Address of Operand + 1	1 1 1 0 0	Op Code Address of Operand Irrelevant Data (Note 1) Register Data (High Order Byte) Register Data (Low Order Byte)
Indexed						
JMP	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Index Register Index Register Plus Offset (w/o Carry)	1 1 1 1	Op Code Offset Irrelevant Data (Note 1) Irrelevant Data (Note 1)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	5	1 2 3 4 5	1 1 0 0 1	Op Code Address Op Code Address + 1 Index Register Index Register Plus Offset (w/o Carry) Index Register Plus Offset	1 1 1 1 1	Op Code Offset Irrelevant Data (Note 1) Irrelevant Data (Note 1) Operand Data
CPX LDS LDX	6	1 2 3 4 5 6	1 1 0 0 1 1	Op Code Address Op Code Address + 1 Index Register Index Register Plus Offset (w/o Carry) Index Register Plus Offset Index Register Plus Offset + 1	1 1 1 1 1 1	Op Code Offset Irrelevant Data (Note 1) Irrelevant Data (Note 1) Operand Data (High Order Byte) Operand Data (Low Order Byte)

Table 8 Operation Summary (Cont.)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/W Line	Data Bus
Indexed (Continued)						
STA	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Current Operand Data
		6	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		7	1/0 (Note 3)	Index Register Plus Offset	0	New Operand Data (Note 3)
STS STX	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data (High Order Byte)
		7	1	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
JSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (High Order Byte)
		5	1	Stack Pointer — 1	0	Return Address (Low Order Byte)
		6	0	Stack Pointer — 2	1	Irrelevant Data (Note 1)
		7	0	Index Register	1	Irrelevant Data (Note 1)
		8	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
Extended						
JMP	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	1	Op Code Address + 2	1	Jump Address (LLow Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data
CPX LDS LDX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data (High Order Byte)
		5	1	Address of Operand + 1	1	Operand Data (High Order Byte)
STA A STA B	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	1	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	0	Operand Destination Address	1	Irrelevant Data (Note 1)
		5	1	Operand Destination Address	0	Data from Accumulator
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Current Operand Data
		5	0	Address of Operand	1	Irrelevant Data (Note 1)
		6	1/0 (Note 3)	Address of Operand	0	New Operand Data (Note 3)

Table 8 Operation Summary (Cont.)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/W Line	Data Bus
Extended (Continued)						
STS STX	6	1 2 3 4 5 6	1 1 1 0 1 1	Op Code Address Op Code Address + 1 Op Code Address + 2 Address of Operand Address of Operand Address of Operand + 1	1 1 1 1 0 0	Op Code Address of Operand (High Order Byte) Address of Operand (Low Order Byte) Irrelevant Data (Note 1) Operand Data (High Order Byte) Operand Data (Low Order Byte)
JSR	9	1 2 3 4 5 6 7 8 9	1 1 1 1 1 1 0 0 1	Op Code Address Op Code Address + 1 Op Code Address + 2 Subroutine Starting Address Stack Pointer Stack Pointer — 1 Stack Pointer — 2 Op Code Address + 2 Op Code Address + 2	1 1 1 1 0 0 1 1 1	Op Code Address of Subroutine (High Order Byte) Address of Subroutine (Low Order Byte) Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte) Irrelevant Data (Note 1) Irrelevant Data (Note 1) Address of Subroutine (Low Order Byte)
Inherent						
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	1 1	Op Code Address Op Code Address + 1	1 1	Op Code Op Code of Next Instruction
DES DEX INS INX	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Previous Register Contents New Register Contents	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data (Note 1) Irrelevant Data (Note 1)
PSH	4	1 2 3 4	1 1 1 0	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer — 1	1 1 0 1	Op Code Op Code of Next Instruction Accumulator Data Accumulator Data
PUL	4	1 2 3 4	1 1 0 1	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data (Note 1) Operand Data from Stack
TSX	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Stack Pointer New Index Register	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data (Note 1) Irrelevant Data (Note 1)
TXS	4	1 2 3 4	1 1 0 0	Op Code Address Op Code Address + 1 Index Register New Stack Pointer	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data Irrelevant Data
RTS	5	1 2 3 4 5	1 1 0 1 1	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Op Code Irrelevant Data (Note 2) Irrelevant Data (Note 1) Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)

Table 8 Operation Summary (Cont.)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/W Line	Data Bus
Inherent (Continued)						
WAI	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer — 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer — 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer — 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer — 4	0	Contents of Accumulator A
		8	1	Stack Pointer — 5	0	Contents of Accumulator B
		9	1	Stack Pointer — 6 (Note 4)	1	Contents of Cond. Code Register
RTI	10	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 2)
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Contents of Cond. Code Register from Stack
		5	1	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	1	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	1	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	1	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	1	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	1	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 1)
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer — 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer — 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer — 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer — 4	0	Contents of Accumulator A
		8	1	Stack Pointer — 5	0	Contents of Accumulator B
		9	1	Stack Pointer — 6	0	Contents of Cond. Code Register
		10	0	Stack Pointer — 7	1	Irrelevant Data (Note 1)
		11	1	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	1	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
Relative						
BCC BHI BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMI BVS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Op Code Address + 2	1	Irrelevant Data (Note 1)
		4	0	Branch Address	1	Irrelevant Data (Note 1)
BSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Return Address of Main Program	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer — 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer — 2	1	Irrelevant Data (Note 1)
		7	0	Return Address of Main Program	1	Irrelevant Data (Note 1)
		8	0	Subroutine Address	1	Irrelevant Data (Note 1)

Notes:
1. If device that is addressed during this cycle uses VMA, the data bus goes to the high-impedance 3-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the data bus.
2. Data is ignored by the MPU.
3. For TST, VMA = 0 and operand data does not change.
4. Most significant byte of address bus = most significant byte of address of BSR instruction, and least significant byte of address bus = least significant byte of subroutine address.

DC Characteristics

Table 9 contains the dc characteristics of the F6802/F6882.

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Voltage of any Pin Relative to GND	−0.3 V,	+7.0 V
Storage Temperature	−55°C,	+150°C
Power Dissipation	1.5 W	
Thermal Resistance, θ_{JA}		
(Plastic Package)	1W	
(CER-DIP Package)	55°C/W	

Table 9 DC Characteristics (V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = 0 to 70°C unless otherwise noted)

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
V_{IH}	Input High Voltage				Vdc	
	Logic EXtal	V_{SS} + 2.0	—	V_{CC}		
	Logic $\overline{Reset}$	V_{SS} + 4.0	—	V_{CC}		
V_{IL}	Input Low Voltage				Vdc	
	Logic Extal, $\overline{Reset}$	V_{SS} − 0.3	—	V_{SS} + 0.8		
I_{IN}	Input Leakage Current				μAdc	
	Logic	—	1.0	2.5		V_{IN} = 0 to 5.25 V, V_{CC} = Max
V_{OH}	Output High Voltage				Vdc	
	D_0-D_7	V_{SS} + 2.4	—	—		I_{LOAD} = −206 μAdc, V_{CC} = Min
	A_0-A_{15}, R/$\overline{W}$, VMA, E	V_{SS} + 2.4	—	—		I_{LOAD} = −145 μAdc, V_{CC} = Min
	BA	V_{SS} + 2.4	—	—		I_{LOAD} = −100 μAdc, V_{CC} = Min
V_{OL}	Output Low Voltage	—	—	V_{SS} + 0.4	Vdc	I_{LOAD} = 1.0 mAdc, V_{CC} = Min
P_D*	Power Dissipation	—	0.600	1.2	W	
	V_{CC} Standby				Vdc	
V_{SBB}	Power Down	4.0	—	5.25		
V_{SB}	Power Up	4.75	—	5.25		
I_{SBB}	Standby Current				mA	
	F6802	—	—	8.0		
	F6882	—	—	3.0		
	Capacitance				pF	
C_{IN}	D_0-D_7	—	—	12.5		
	Logic Inputs EXtal	—	6	10		V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	A_0-A_{15}, R/$\overline{W}$, VMA	—	—	12		

*In power-down mode, maximum power dissipation is less than 42 mW.
Capacitances are periodically sampled rather than 100% tested.

Timing Characteristics

Tables 10 and 11 contain timing characteristics information.

Table 10 Frequency Characteristics

Symbol	Characteristic	F6802 Min	F6802 Max	F680A02 Min	F680A02 Max	Unit
f_o	Frequency of Operation	0.1	1.0	0.1	1.5	MHz
f_{XTAL}	Crystal Frequency	1.0	4.0	1.0	1.5	MHz
$4xf_o$	External Oscillator Frequency	0.4	4.0	0.4	6.0	MHz
t_{CYC}	Cycle Time	1.0	10	0.666	10	μs
t_{PWEH}	Clock Pulse Width E High	450	9500	280	9700	ns
t_{PWEL}	Clock Pulse Width E Low	450	5000	280	5000	ns
t_R, t_F	Fall Time	—	25	—	25	ns
t_{rc}	Crystal Oscillator Startup Time	100	—	100	—	ms

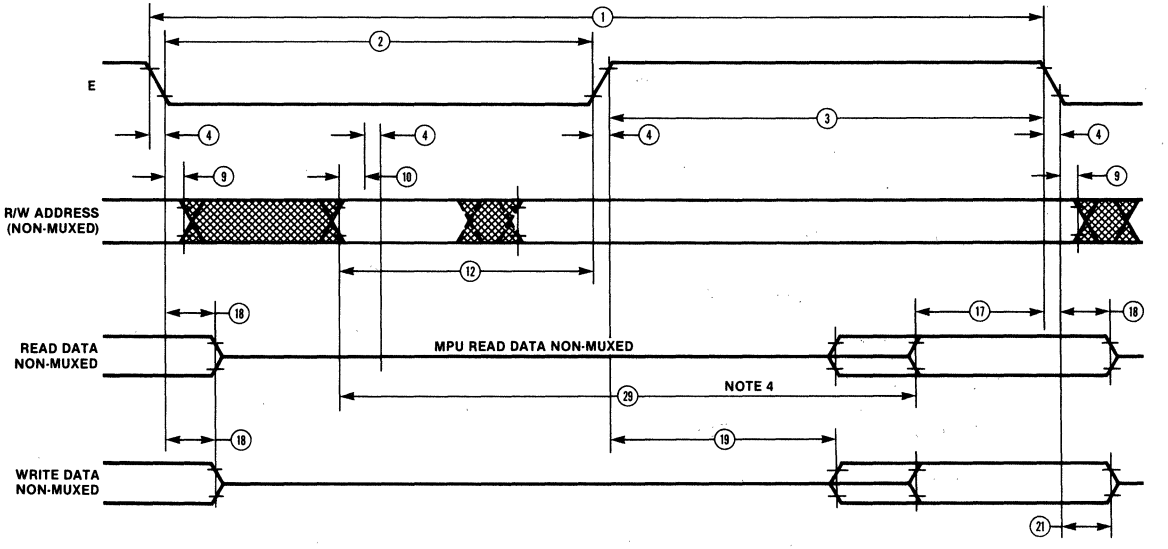
Table 11 Read/Write Timing

Symbol	Characteristic	F6802 Min	F6802 Max	F680A02 Min	F680A02 Max	Unit
t_{AD}	Address Delay	—	270	—	220	ns
t_{AV1}	Address Delay (Internal RAM	—	270	—	240	ns
t_{AV2}	Read Access Time Useable by Peripheral @ 1 MHz $t_{ACC} = t_{CYC} - t_{AD} + t_{DSR} + t_F$	—	605	—	310	
t_{DSR}	Data Setup Time (Read)	100	—	70	—	ns
t_{DHR}	Input Data Hold Time	10	—	10	—	ns
t_{DHW}	Output Data Hold Time	30	—	20	—	ns
t_{AH}	Address Hold Time (Address, R/W, VMA)	20	—	20	—	ns
t_{DDW}	Data Delay Time (Write)	—	225	—	170	ns
	Processor Controls					
t_{PCS}	Processor Control Setup Time	200	—	140	—	
t_{PCR}, t_{PCF}	Processor Control Rise and Fall Time (Does Not Apply to RESET) (Measured between 0.8 V and 2.0 V)	—	100	—	100	ns

Note
If programs are not executed from on-board RAM, TAV1 applies. If programs are to be stored and executed from on-board RAM, TAV2 applies. For normal data storage in the on-board RAM, this extended delay does not apply. Programs cannot be executed from on-board RAM when using A parts (F68A02, F68A08). One-board RAM can be used for data storage with all parts.

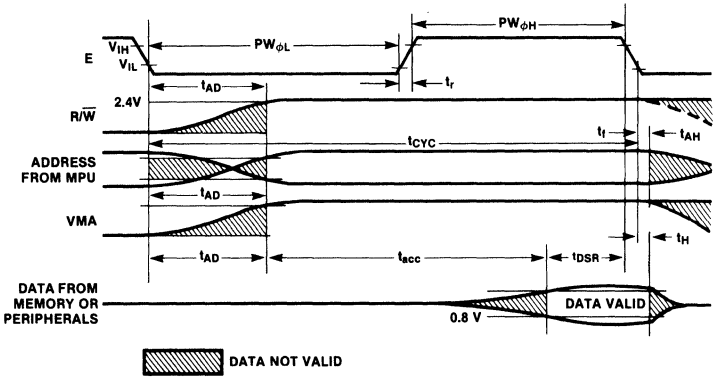
Bus Timing Characteristics

Symbol	Parameter	F6802NS F6802 F6808 Min	F6802NS F6802 F6808 Max	F68A02 F68A08 Min	F68A02 F68A08 Max	Unit
t_{CYC}	(1) Cycle Time	1.0	10	0.667	10	μs
PW_{EL}	(2) Pulse Width, E Low	450	5000	280	5000	ns
PW_{EH}	(3) Pulse Width, E High	450	9500	280	9700	ns
t_r, t_f	(4) Clock Rise and Fall Time	—	25	—	25	ns
t_{AH}	(9) Address Hold Time	20	—	20	—	ns
t_{AV1} t_{AV2}	(12) Non-Muxed Address Valid Time to E	160 —	— 270	100 —	— —	ns
t_{DSR}	(17) Read Data Setup Time	100	—	70	—	ns
t_{DHR}	(18) Read Data Hold Time	10	—	10	—	ns
t_{DDW}	(19) Write Data Delay Time	—	225	—	170	ns
t_{DHW}	(21) Wrote Data Hold Time	30	—	20	—	ns
t_{ACC}	(29) Usable Access Time (See Note 4)	605	—	310	—	ns



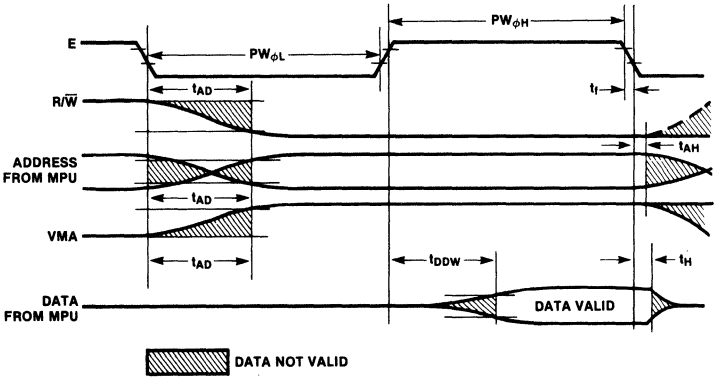
Notes
1. Voltage levels shown are $V_L \leq 0.4$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise noted.
3. All electricals shown for the F6802 apply to the F6802NS and F6808, unless otherwise noted.
4. Usable access time is computed by 12 + 3 + 4 − 17.

Fig. 15 Read Data from Memory or Peripherals



Note
Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

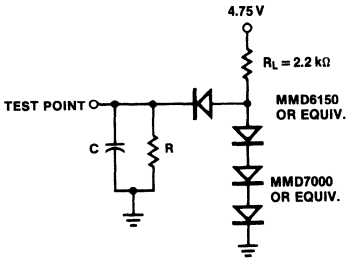
Fig. 16 Write Data in Memory or Peripherals



Note
Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

Fig. 17 Bus Timing Test Load

C = 130 pF FOR D_0-D_7, E
= 90 pF FOR A_0-A_{15}, R/$\overline{W}$, AND VMA
= 30 pF FOR BA
R = 11.7 kΩ FOR D_0-D_7, E
= 16.5 kΩ FOR A_0-A_{15}, R/$\overline{W}$, AND VMA
= 24 kΩ FOR BA



Ordering Information

Order Code	Speed	Temperature Range
F6802P, S	1.0 MHz	0°C to +70°C
F6882P,S	1.0 MHz	0°C to +70°C
F6802CP, CS	1.0 MHz	−40°C to +85°C
F68A02P, S	1.5 MHz	0°C to +70°C
F68A02CP, CS	1.5 MHz	−40°C to +85°C

P = Plastic package
S = Ceramic package

FAIRCHILD
A Schlumberger Company

F6809/F68A09/F68B09
Central Processing Unit

Advance Product Information | Microprocessor Product

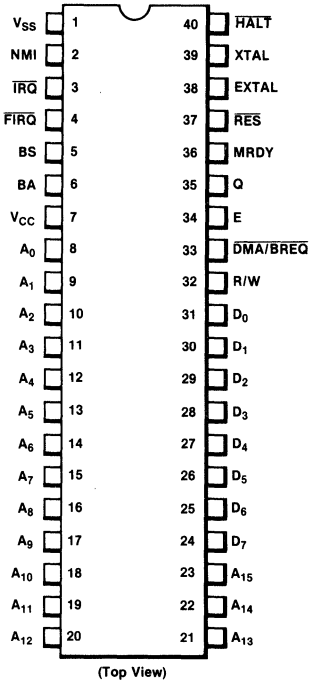
Description

The Fairchild F6809 8-bit Microprocessor is an advanced, high-performance member of the F6800 family. It offers greater throughput, improved byte efficiency, and increased adaptability to various software disciplines, including position-independent code, re-entrancy, recursion, block structuring, and high-level language generation. The F6809 is compatible with all F6800 peripheral devices and is upward source code compatible with F6800-series microprocessors. The device is available in three frequency ranges: 1.0 MHz (F6809), 1.5 MHz (F68A09), and 2.0 MHz (F68B09).

Architectural improvements, such as additional 16-bit registers and dual 8-bit data paths, allow for powerful enhancements to the instruction set and addressing capabilities.

- **Compatible with Entire F6800 Family**
  - **—Hardware Interfaces with all F6800 Peripherals**
  - **—Software Has Upward-Compatible Instruction Set and Addressing Modes**
- **Two 16-Bit Index Registers**
- **Two Indexable 16-Bit Stack Pointers**
- **Two 8-Bit Accumulators Can Be Concatenated to Form One 16-Bit Accumulator**
- **Direct Page Register Allows Direct Addressing Throughout Memory Map**
- **Single +5 V Supply**
- **On-Chip Oscillator**
- **MRDY Input Extends Data Access Time for Use with Slow Memory**
- **$\overline{\text{DMA}}$/$\overline{\text{BREQ}}$ Allows Access to Bus for DMA and Memory Refresh**
- **Fast Interrupt Request ($\overline{\text{FIRQ}}$) Stacks Only Program Counter and Condition Code Register**
- **Interrupt Acknowledge Output Allows Vectoring by Device**
- **Sync Acknowledge Output Allows for Synchronization to External Event**
- **16-Bit Arithmetic (ADD, SUBTRACT, COMPARE, LOAD, STORE)**
- **8 × 8 Unsigned Multiply**
- **Transfer/Exchange all Registers**
- **Push/Pull all Registers**
- **Ten Addressing Modes**
- **Expanded Indexed Addressing, Accumulator or up to 16-Bit Offset, Auto-Increment/Decrement by One or Two**
- **True Indirect Addressing**
- **Load-Effective Address**

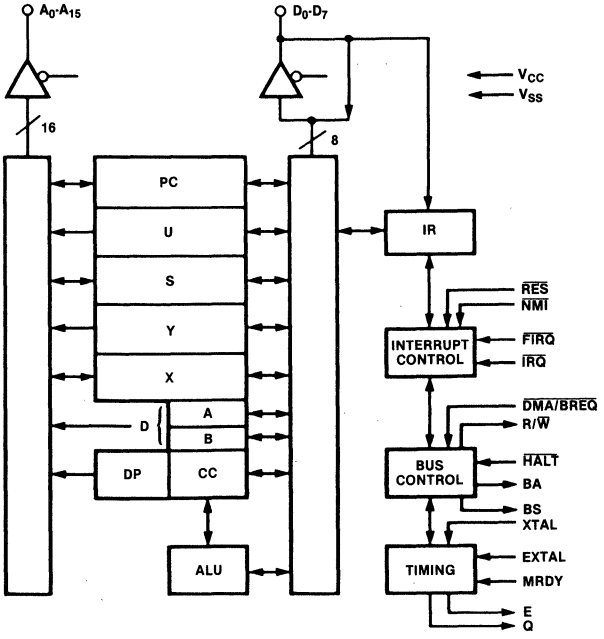
Connection Diagram



(Top View)

5

Block Diagram



FAIRCHILD
A Schlumberger Company

F6809E/F68A09E/F68B09E
Central Processing Unit

Advance Product Information | Microprocessor Product

Description

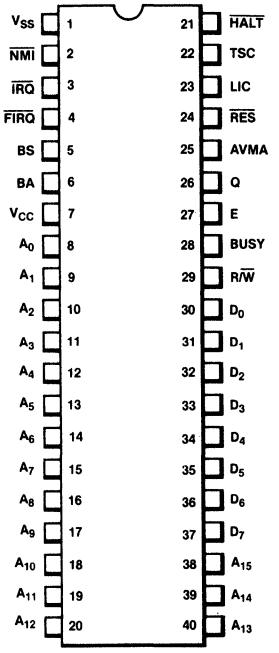
The Fairchild F6809E 8-bit microprocessor is an advanced, high-performance member of the F6800 family. It offers greater throughput, improved byte efficiency, and increased adjustability to various software disciplines, including position-independent code, re-entrancy, recursion, block structuring, and high-level language generation. The F6809E is compatible with all F6800 peripheral devices and is upward source code compatible with F6800-series microprocessors. It is available in three frequency ranges: 1.0 MHz (F6809E); 1.5 MHz (F68A09E); 2.0 MHz (F68B09E).

Architectural improvements, such as additional 16-bit registers and dual 8-bit data paths, allow for powerful enhancements to the instruction set and addressing capabilities.

External clock inputs are provided on the F6809E to allow synchronization with peripherals, systems, or other MPUs.

- **Compatible with Entire F6800 Family**
  - **Hardware Interfaces with All F6800 Peripherals**
  - **Software Has Upward-Compatible Instruction Set and Addressing Modes**
- **Two 16-Bit Index Registers**
- **Two Indexable 16-Bit Stack Registers**
- **Two 8-bit Accumulators Can Be Connected to Form One 16-Bit Accumulator**
- **Direct Page Register Allows Direct Addressing Throughout Memory Map**
- **External E and Q Clock Inputs Allow Synchronization**
- **TSC Input Controls Internal Bus Buffers**
- **LIC Output Indicates Opcode Fetch**
- **AVMA Output Allows Efficient Use of Common Resources in a Multiprocessor System**
- **Fast Interrupt Request Input Stacks Only Program Counter and Condition Code Register**
- **Interrupt Acknowledge Output Allows Vectoring by Device**
- **Sync Acknowledge Output Allows for Synchronization to External Event**
- **Single Bus-Cycle Reset**
- **Single + 5 V Supply**
- **Early Address Valid Allows Use with Slower Memories**
- **Early Write-Data for Dynamic Memories**
- **10 Addressing Modes**
- **True Indirect Addressing**
- **Expanded Indexed Addressing**
- **1464 Instructions with Unique Addressing Modes**
- **Recognizes 8x8 Unsigned Multiply**
- **16-Bit Arithmetic**

Connection Diagram



5

Table 1 DC Characteristics

			F6809E			F68A09E			F68B09E				
Symbol	**Characteristic**		**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Unit**	**Test Conditions**
V_{IH} V_{IHR} V_{IHC}	Input High Voltage	Logic, Q	2.2	—	V_{CC}	2.2	—	V_{CC}	2.2	—	V_{CC}	V	
		$\overline{RES}$	4.0	—	V_{CC}	4.0	—	V_{CC}	4.0	—	V_{CC}	V	
		E	V_{CC} −0.75	—	V_{CC} +0.3	V_{CC} −0.75	—	V_{CC} +0.3	V_{CC} −0.75	—	V_{CC} +0.3	V	
V_{IL} V_{ILC}	Input Low Voltage	Logic Q, $\overline{RES}$	−0.2	—	0.8	−0.2	—	0.8	−0.2	—	0.8	V	
		E	−0.3	—	0.4	−0.3	—	0.4	−0.3	—	0.4	V	
I_{IN}	Input Leakage Current	Logic Q, $\overline{RES}$	−2.5	—	2.5	−2.5	—	2.5	−2.5	—	2.5	μA	V_{IN} = 0 - 5.25 V; V_{CC} = Max
		E	−100	—	100	−100	—	100	−100	—	100	μA	
V_{OH}	Output High Voltage	D_0-D_1	2.4	—	—	2.4	—	—	2.4	—	—	V	I_{LOAD} = −205 μA V_{CC} = Min
		A_0-A_{15}, R/$\overline{W}$	2.4	—	—	2.4	—	—	2.4	—	—	V	I_{LOAD} = −145 μA V_{CC} = Min
		BA, BS, LIC, AVMA, BUSY	2.4	—	—	2.4	—	—	2.4	—	—	V	I_{LOAD} = −100 μA V_{CC} = Min
V_{OL}	Output Low Voltage		—	—	0.5	—	—	0.5	—	—	0.5	V	I_{LOAD} = 2 mA; V_{CC} = Min
P_D	Power Dissipation		—	—	1.0	—	—	1.0	—	—	1.0	W	
C_{IN}	Input Capacitance	D_0-D_7, Logic Input, Q, $\overline{RES}$	—	10	15	—	10	15	—	10	15	pF	V_{IN} = 0 V; T_A = 25 °C; F = 1 MHz
		E	—	30	50	—	30	50	—	30	50	pF	
C_{OUT}	Output Capacitance	A_0-A_{15}, R/$\overline{W}$, BA, BS, LIC, AVMA, BUSY	—	10	15	—	10	15	—	10	15	pF	V_{IN} = 0 V; T_A = 25 °C; F = 1 MHz
F	Frequency of Operation (E, Q)		0.1	—	1.0	0.1	—	1.5	0.1	—	2.0	MHz	
I_{TSI}	Three-State (Off State) Input Current	D_0-D_7	−10	—	10	−10	—	10	−10	—	10	μA	V_{IN} = 0.4 − 2.4 V V_{CC} = Max
		A_0-A_{15}, R/$\overline{W}$	−100	—	100	−100	—	100	−100	—	100	μA	

V_{CC} = 5.0 V ±5%; V_{SS} = 0 V; T_A = −20 °C - +70 °C, unless otherwise noted.

Functional Description

During normal operation, the F6809E fetches an instruction from memory and executes the requested function. This sequence begins following a Reset ($\overline{RES}$) input and is repeated indefinitely unless altered by a special instruction or hardware occurence. Software instructions that alter normal operation are: $\overline{SWI}$, SWI2, SWI3, CWAI, RTI, and SYNC. An interrupt or $\overline{HALT}$ input can modify normal instruction execution.

Instruction Set

The instruction set of the F6809E is similar to that of the F6800 and is upward-compatible at the source code level. The number of opcodes has been reduced to 59 but, because of the expanded architecture and additional addressing modes, the number of available opcodes (with different addressing modes) is 1464.

Some of the new instructions are:

1. PSH6/PSH5, which are push instructions that have the capability of pushing onto either the user stack (U) or hardware stack (S) any register or set of registers with a single instruction.

2. PUL6/PUL5, which are pull instructions having the same capabilities as the push instructions, in reverse order.

3. TFR/EXG, which allow any register to be transferred to or exchanged with another register of like size.

4. LEAX/LEAY/LEAU/LEAS, which calculate the effective address used in an indexed instruction and store that effective value, rather than the data at that address, in a pointer register.

5. MUL, which multiplies the unsigned binary numbers in the A and B accumulators and places the unsigned result into the 16-bit D accumulator.

Addressing Modes

The F6809E has 59 basic instructions; however, it recognizes 1464 different variations of instructions and addressing modes. The addressing modes available on the F6809E are:

1. Implied (includes accumulator)
2. Immediate
3. Extended
4. Extended indirect
5. Direct
6. Register
7. Indexed
   a. Zero-offset
   b. Constant offset
   c. Accumulator offset
   d. Auto increment/decrement
8. Indexed indirect
9. Relative
10. Program counter relative

5

DC Characteristics

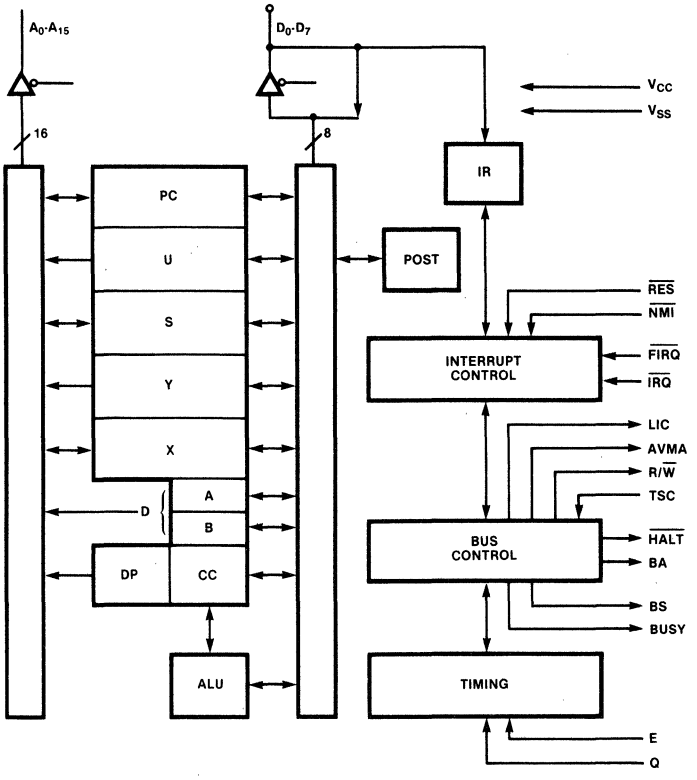
Table 1 describes the dc characteristics of the F6809E.

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this document, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Storage Temperature Range	−55°C, +150°C
Operating Temperature Range	−20°C, +75°C
V_{CC}	−0.3 V, +7.0 V
Input Voltage	−0.3 V, +7.0 V

Block Diagram



FAIRCHILD
A Schlumberger Company

F6810/F68A10/F68B10
128 x 8-Bit Static Random Access Memory

Microprocessor Product

Description

The F6810 128 x 8-bit static RAM is a byte-organized memory designed for use in bus-organized systems. Fabricated with n-channel, silicon-gate technology, the device is available in three frequency ranges: 1.0 MHz (F6810), 1.5 MHz (F68A10), 2.0 MHz (F68B10). The device, which operates from a single power supply, is compatible with TTL and DTL; it needs no clocks or refreshing because of its static operation.

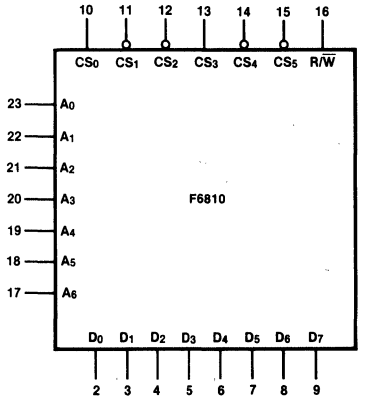
The memory is compatible with the F6800 microcomputer family, providing random storage in byte increments. Memory expansion is provided through multiple chip select inputs.

- **Organized as 128 Bytes of 8 Bits**
- **Static Operation**
- **Bidirectional 3-State Data Input/Output**
- **Six Chip Select Inputs (Four Active LOW, Two Active HIGH)**
- **Single +5 V Power Supply**
- **TTL-Compatible**
- **Maximum Access Time:**
  450 ns for F6810
  360 ns for F68A10
  250 ns for F68B10

Pin Names

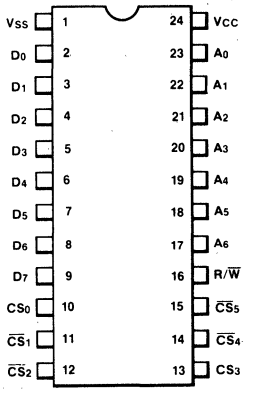
D_0-D_7	Bidirectional Data Bus
A_0-A_6	Address Inputs
CS_0-$\overline{CS}_5$	Chip Select Inputs
R/$\overline{W}$	Read/Write Input

Logic Symbol



V_{CC} = Pin 24
V_{SS} = Pin 1

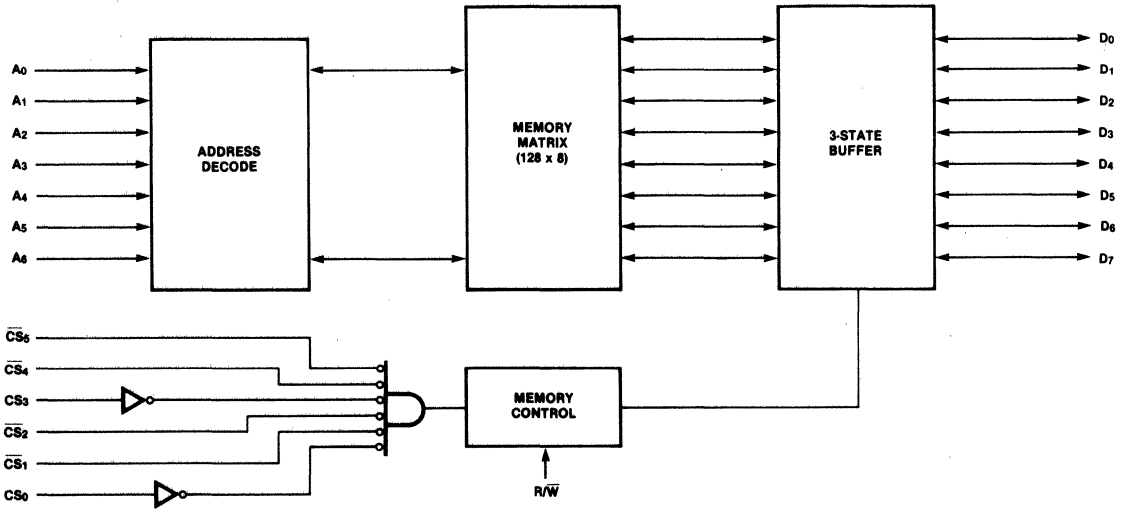
Connection Diagram
24-Pin DIP



(Top View)

5

Block Diagram



Signal Function Descriptions

Mnemonic	Pin No.	Name	Description
Bus Handshake			
A_0-A_6	17-23	Address Bus	Input signal lines containing address to which data is to be written or from which data is to be read
D_0-D_7	2-9	Data Bus	Bidirectional input/output signal lines over which data is read from or written to the device
Chip Control			
CS_0-$\overline{CS}_5$	10-15	Chip Select	Input signal lines that prepare the device for a read or write operation
R/$\overline{W}$	16	Read/Write	Input signal lines that selects a chip read or write operation; a HIGH selects memory read, and a LOW selects memory write
Supply			
V_{SS}	1	Ground	Ground for supply and signals
V_{CC}	24	Supply	+5 V supply voltage

Absolute Maximum Ratings

Supply Voltage	−0.3 V, +7.0 V
Input Voltage	−0.3 V, +7.0 V
Operating Temperature - T_L, T_H	
F6810, F68A10, F68B10	0 °C, +70 °C
F6810C, F68A10C	−40 °C, +85 °C
F6810DM	−55 °C, +125 °C
Storage Temperature Range	−65 °C, +150 °C
Thermal Resistance - θ_{JA}	82.5 °C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages.

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Recommended Operating Conditions

Symbol	Characteristic	Min	Typ	Max	Unit
V_{CC}	Supply Voltage	4.75	5.0	5.25	V
V_{IH}	Input HIGH Voltage	2.0		5.25	V
V_{IL}	Input LOW Voltage	−0.3		0.8	V

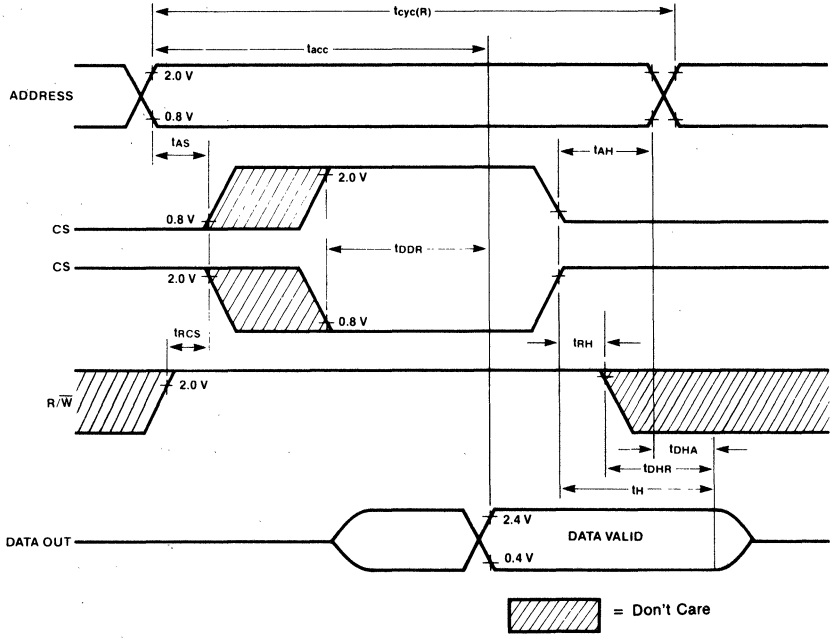
DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted.

Symbol	Characteristic	Min	Typ	Max	Unit	Conditions
I_{IN}	Input Current (A_n, R/$\overline{W}$, CS_n, $\overline{CS}_n$)			2.5	μA	V_{IN} = 0 to 5.25 V
V_{OH}	Output HIGH Voltage	2.4			V	I_{OH} = −205 μA
V_{OL}	Output LOW Voltage			0.4	V	I_{OL} = 1.6 mA
I_{LO}	Output Leakage Current, 3-State			10	μA	$\overline{CS}$ = 0.8 V or $\overline{CS}$ = 2.0 V, V_O = 0.4 V to 2.4 V
I_{CC}	Supply Current F6810 F68A10, F68B10			80 100	mA	V_{CC} = 5.25 V, all other pins grounded, T_A = 0°C
C_{IN}	Input Capacitance			7.5	pF	f = 1.0 MHz, T_A = 25°C
C_{OUT}	Output Capacitance			12.5	pF	

Bus Timing Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted.

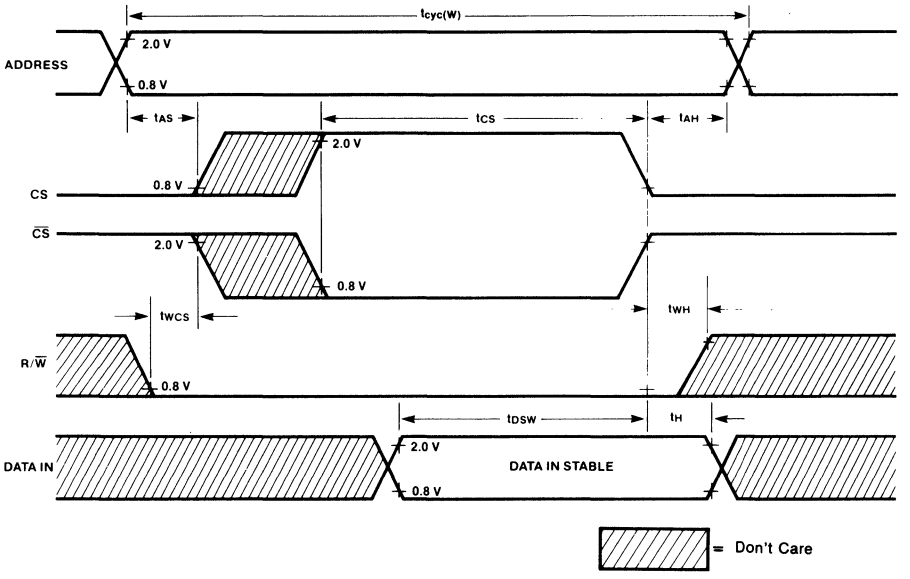
Symbol	Characteristic	F6810		F68A10		F68B10		Unit
		Min	Max	Min	Max	Min	Max	
Read *(Figure 1)*								
$t_{cyc(R)}$	Read Cycle Time	450		360		250		ns
t_{acc}	Access Time		450		360		250	ns
t_{AS}	Address Set-up Time	20		20		20		ns
t_{AH}	Address Hold Time	0		0		0		ns
t_{DDR}	Data Delay Time (Read)		230		220		180	ns
t_{RCS}	Read-to-Select Delay Time	0		0		0		ns
t_{DHA}	Data Hold from Address	10		10		10		ns
t_H	Output Hold Time	10		10		10		ns
t_{DHR}	Data Hold from Read	10	80	10	60	10	60	ns
t_{RH}	Read Hold from Chip Select	0		0		0		ns
Write *(Figure 2)*								
$t_{cyc(W)}$	Write Cycle Time	450		360		250		ns
t_{AS}	Address Set-up Time	20		20		20		ns
t_{AH}	Address Hold Time	0		0		0		ns
t_{CS}	Chip Select Pulse Width	300		250		210		ns
t_{WCS}	Write-to-Chip Select Delay Time	0		0		0		ns
t_{DSW}	Data Set-up Time (Write)	190		80		60		ns
t_H	Input Hold Time	10		10		10		ns
t_{WH}	Write Hold from Chip Select	0		0		0		ns

Fig. 1 Read Cycle Timing



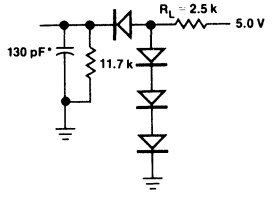
Note
CS and $\overline{CS}$ can be enabled for consecutive read cycles, provided R/$\overline{W}$ remains at V_{IH}.

Fig. 2 Write Cycle Timing



Note
CS and $\overline{CS}$ can be enabled for consecutive write cycles, provided R/$\overline{W}$ is strobed to V_{IH} before or coincident with the address change, and remains HIGH for time t_{AS}.

Fig. 3 Output Load



*Includes jig capacitance

Timing Conditions
The conditions under which the timing characteristics have been determined are as follows:

Input Pulse Levels	0.8 V to 2.0 V
Input Rise and Fall Times	20 ns
Output Load	See *Figure 3*
V_{CC}	5.0 V ±5%
V_{SS}	0
T_A	T_L to T_H, unless otherwise noted

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6810P,S	0°C to 70°C
	F6810CP,CS	−40°C to +85°C
	F6810DM	−55°C to +125°C
1.5 MHz	F68A10P,S	0°C to +70°C
	F68A10CP,CS	−40°C to +85°C
2.0 MHz	F68B10DM	−55°C to +125°C
	F68B10P,S	0°C to +70°C

P = Plastic package, S = Ceramic package

FAIRCHILD
A Schlumberger Company

F6820
Peripheral Interface Adapter (PIA)

Microprocessor Product

Description

The F6820* Peripheral Interface Adapter (PIA) provides the universal means of interfacing peripheral equipment to the F6800 Microprocessing Unit (MPU). This device is capable of interfacing the MPU to peripherals through two 8-bit bidirectional peripheral data buses and four control lines. No external logic is required for interfacing to most peripheral devices.

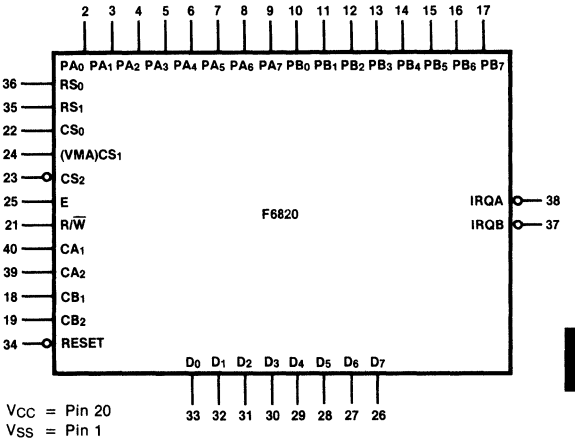
The functional configuration of the PIA is programmed by the MPU during system initialization. Each of the peripheral data lines can be programmed to act as an input or output, and each of the four control/interrupt lines may be programmed for one of several control modes. This allows a high degree of flexibility in the overall operation of the interface.

- **8-Bit Bidirectional Data Bus for Communication with the MPU**
- **Two Bidirectional 8-Bit Buses for Interface to Peripherals**
- **Two Programmable Control Registers**
- **Two Programmable Data Direction Registers**
- **Four Individually Controlled Interrupt Input Lines, Two Usable as Peripheral Control Outputs**
- **Handshake Control Logic for Input and Output Peripheral Operation**
- **High-Impedance 3-State and Direct Transistor Drive Peripheral Lines**
- **Program-Controlled Interrupt and Interrupt Disable Capability**
- **CMOS Drive Capability on Side A Peripheral Lines**

Pin Names

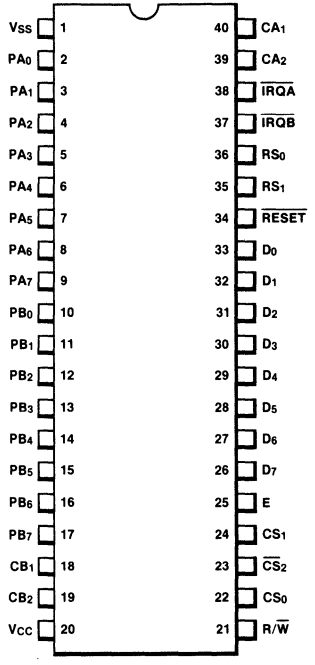
D_0-D_7	Bidirectional Data Bus
PA_0-PA_7	Bidirectional Peripheral Data Bus A
PB_0-PB_7	Bidirectional Peripheral Data Bus B
CS_0, CS_1, $\overline{CS_2}$	Chip Select Inputs
RS_0, RS_1	Register Select Inputs
E	Enable Input
R/$\overline{W}$	Read/Write Input
$\overline{RESET}$	Reset Input
CA_1, CB_1	Interrupt Control Inputs
CA_2, CB_2	Programmable Interrupt Control Input or Peripheral Control Output
$\overline{IRQA}$, $\overline{IRQB}$	Interrupt Request Outputs

Logic Symbol



V_{CC} = Pin 20
V_{SS} = Pin 1

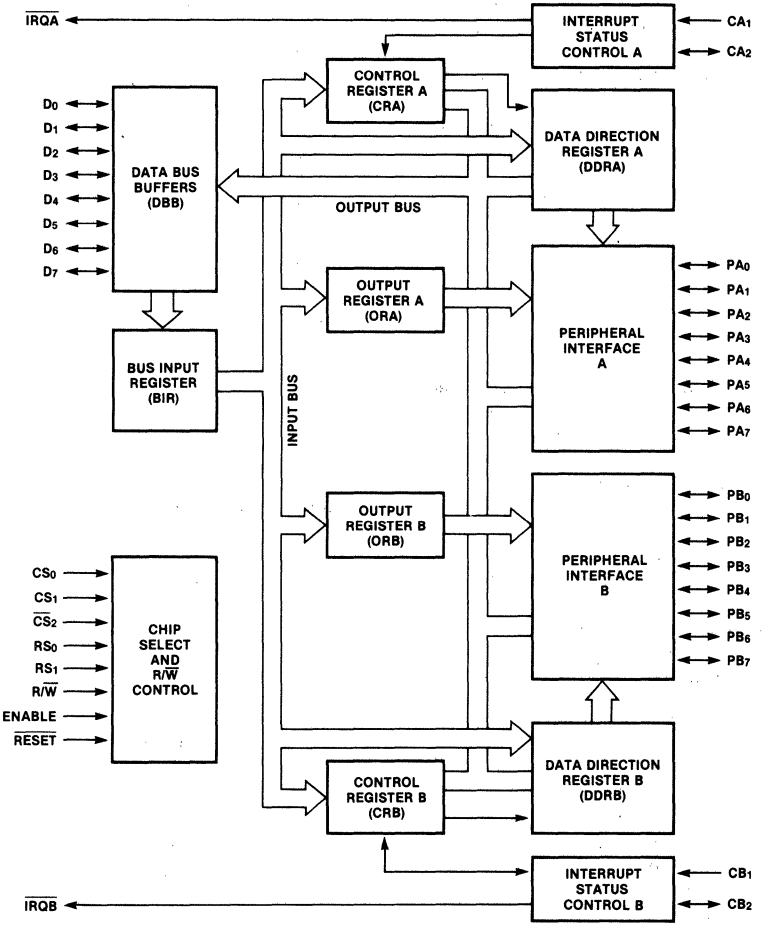
Connection Diagram
40-Pin DIP



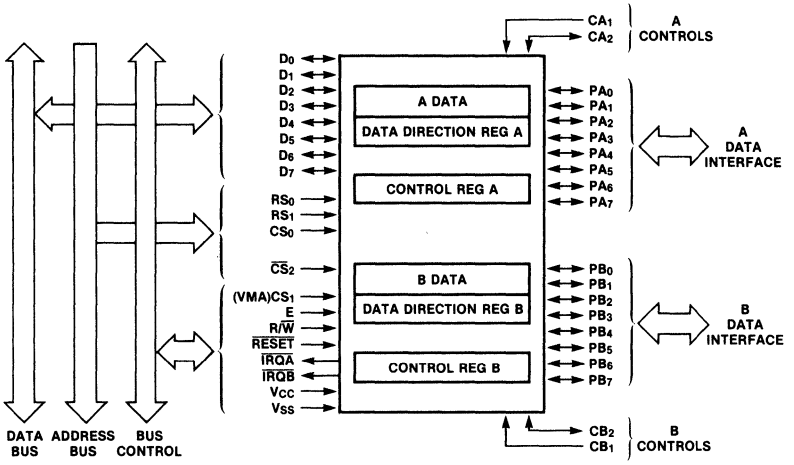
(Top View)

* Not recommended for new designs.

Block Diagram



F6820 PIA Bus Interface



5

Functional Description

PIA Interface Signals for MPU

The PIA interfaces to the F6800 MPU with an 8-bit bidirectional data bus, three chip select lines, two register select lines, two interrupt request lines, a read/write line, an enable line, and a reset line. These signals, in conjunction with the F6800 VMA output, permit the MPU to have complete control over the PIA. The VMA signal should be utilized in conjunction with an MPU address line into a chip select of the PIA.

PIA Bidirectional Data (D_0-D_7), Pins 26-33 — The bidirectional data lines (D_0-D_7) allow the transfer of data between the MPU and the PIA. The data bus output drivers are 3-state devices that remain in the high-impedance (OFF) state except when the MPU performs a PIA read operation. The read/write line is in the read (HIGH) state when the PIA is selected for a read operation.

PIA Enable (E), Pin 25 — The enable (E) pulse is the only timing signal that is supplied to the PIA. Timing of all other signals is referenced to the leading and trailing edges of the E pulse. This signal is normally a derivative of the F6800 $\phi 2$ clock.

PIA Read/Write (R/$\overline{W}$), Pin 21 — This signal is generated by the MPU to control the direction of data transfers on the data bus. A LOW on the PIA read/write line enables the input buffers, and data is transferred from the MPU to the PIA on the E signal if the device has been selected. A HIGH on the read/write line sets up the PIA for a transfer of data to the bus. The PIA output buffers are enabled when the proper address and the E pulse are present.

$\overline{\text{RESET}}$, Pin 34 — The active-LOW $\overline{\text{RESET}}$ line is used to reset all register bits in the PIA to a logic 0 (LOW). This line can be used as a power-on reset and as a master reset during system operation.

PIA Chip Select (CS_0, CS_1, and $\overline{CS_2}$), Pins 22-24 — These three input signals are used to select the PIA. CS_0 and CS_1 must be HIGH and $\overline{CS_2}$ must be LOW for selection of the device. Data transfers are then performed under the control of the enable and read/write signals. The chip select lines must be stable for the duration of the E pulse. The device is "deselected" when any of the chip selects are in the inactive state.

PIA Register Select (RS_0 and RS_1), Pins 35, 36 — The two register select lines are used to select the various registers inside the PIA. These two lines are used in conjunction with internal control registers to select a particular register that is to be written to or read.

The register and chip select lines should be stable for the duration of the E pulse while in the read or write cycle.

Interrupt Request ($\overline{IRQA}$ and $\overline{IRQB}$), Pins 37, 38 — The active-LOW interrupt request lines act to interrupt the MPU either directly or through interrupt priority circuitry. These lines are open drain (no-load device on the chip). This permits all interrupt request lines to be tied together in a wired-OR configuration.

Each interrupt request line has two internal interrupt flag bits that can cause either line to go LOW. Each flag bit is associated with a particular peripheral interrupt line. Also four interrupt enable bits are provided in the PIA that are used to inhibit a particular interrupt from a peripheral device.

Servicing an interrupt by the MPU is accomplished by a software routine that, on a priority basis, sequentially reads and tests the two control registers in each PIA for interrupt flag bits that are set.

The interrupt flags are cleared (set to 0) as a result of an MPU read peripheral data operation of the corresponding data register. After being cleared, the interrupt flag bit cannot be enabled until the PIA is "deselected"during an E pulse. The E pulse is used to condition the interrupt control lines (CA_1, CA_2, CB_1, CB_2). When these lines are used as interrupt inputs, at least one E pulse must occur from the inactive edge to the active edge of the interrupt input signal to condition the edge sense network. If the interrupt flag has been enabled and the edge sense circuit has been properly conditioned, the interrupt flag is set on the next active transition of the interrupt input pin.

PIA/Peripheral Interface Lines

The PIA provides two 8-bit bidirectional data buses and four interrupt/control lines for interfacing to peripheral devices.

Section A Peripheral Data (PA_0-PA_7), Pins 2-9 — Each of the peripheral data lines is programmed to act as an input or output. This is accomplished by setting a "1" in the corresponding data direction register bit for those lines which are to be outputs. A "0" in a bit of the data direction register causes the corresponding peripheral data line to act as an input. During an MPU read peripheral data operation, the data on peripheral data lines programmed to act as inputs appears directly on the corresponding MPU data bus lines. In the input mode, the internal pull-up resistor on these lines represents a maximum of one standard TTL load.

The data in output register A appears on the data lines that are programmed to be outputs. A logic "1" written into the register causes a HIGH on the corresponding data line, while a "0" results in a LOW. Data in output register A may be read by an MPU read peripheral data A operation when the corresponding lines are programmed as outputs. This data is read properly if the voltage on the peripheral data lines is greater than 2.0 V for a logic "1" output and less than 0.8 V for a logic "0" output. Loading the output lines so that the voltage on these lines does not reach full voltage causes the data transferred into the MPU on a read operation to differ from that contained in the respective bit of output register A.

Section B Peripheral Data (PB_0-PB_7), Pins 10-17 — The peripheral data lines in the B Section of the PIA can be programmed to act as either inputs or outputs in a similar manner to PA_0-PA_7. However, the output buffers driving these lines differ from those driving lines PA_0-PA_7. They have 3-state capability, allowing them to enter a high-impedance state when the peripheral data line is used as an input. In addition, data on the peripheral data lines PB_0-PB_7 is read properly from those lines programmed as outputs even if the voltages are below 2.0 V for a HIGH. As outputs, these lines are compatible with standard TTL and may also be used as a source of up to 1 mA at 1.5 V to drive the base of a transistor switch directly.

Interrupt Input (CA_1 and CB_1), Pins 18, 40 — Interrupt input lines CA_1 and CB_1 are input-only lines that set the interrupt flags of the control registers. The active transition for these signals is also programmed by the two control registers.

Peripheral Control (CA_2), Pin 39 — The peripheral control line CA_2 can be programmed to act as an interrupt input or as a peripheral control output. As an output, this line is compatible with standard TTL; as an input, the internal pull-up resistor on this line represents one standard TTL load. The function of this signal line is programmed with control register A.

Peripheral Control (CB_2), Pin 19 — Peripheral control line CB_2 may also be programmed to act as an interrupt input or peripheral control output. As an input, this line has high input impedance and is compatible with standard TTL. As an output, it is compatible with standard TTL and may also be used as a source of up to

1 mA at 1.5 V to drive the base of a transistor switch directly. This line is programmed by control register B.

Note
It is recommended that the control lines (CA_1, CA_2, CB_1, CB_2) should be held in a logic "1" state when $\overline{RESET}$ is active to prevent setting of corresponding interrupt flags in the control register when $\overline{RESET}$ goes to an inactive state. Subsequent to $\overline{RESET}$ going inactive, a read of the data registers may be used to clear any undesired interrupt flags.

Internal Controls
There are six locations within the PIA accessible to the MPU data bus: two peripheral registers, two data direction registers, and two control registers. Selection of these locations is controlled by the RS_0 and RS_1 inputs together with bit 2 in the control register, as shown in *Table 1.*

Table 1 Internal Addressing

		Control Register Bit		
RS_1	RS_0	CRA-2	CRB-2	Location Selected
0	0	1	X	Peripheral Register A
0	0	0	X	Data Direction Register A
0	1	X	X	Control Register A
1	0	X	1	Peripheral Register B
1	0	X	0	Data Direction Register B
1	1	X	X	Control Register B

X = Don't Care

Initialization
A LOW $\overline{RESET}$ line has the effect of zeroing all PIA registers. This sets PA_0-PA_7, PB_0-PB_7, CA_2 and CB_2 as inputs, with all interrupts disabled. The PIA must be configured during the restart program which follows the reset.

Possible configurations of the data direction and control registers are as follows.

Data Direction Registers (DDRA and DDRB)
The two data direction registers allow the MPU to control the direction of data through each corresponding peripheral data line. A data direction register bit set at "0" configures the corresponding peripheral data line as an input; a "1" results in an output.

Control Registers (CRA and CRB)
The two control registers (CRA and CRB) allow the MPU to control the operation of the four peripheral control lines CA_1, CA_2, CB_1 and CB_2. In addition, they allow the MPU to enable the interrupt lines and monitor the status of the interrupt flags. Bits 0 through 5 of the two registers are written or read by the MPU when the proper chip select and register select signals are applied. Bits 6 and 7 of the two registers are read only and are modified by external interrupts occurring on control lines CA_1, CA_2, CB_1 or CB_2. The format of the control words is shown in *Table 2.*

Table 2 Control Word Format

Bit	CRA	CRB
7	IRQA1	IRQB1
6	IRQA2	IRQB2
5 4 3	CA_2 Control	CB_2 Control
2	DDRA Access	DDRB Access
1 0	CA_1 Control	CB_1 Control

Data Direction Access Control Bit (CRA-2 and CRB-2) — Bit 2 in each control register (CRA and CRB) allows selection of either a peripheral interface register or the data direction register when the proper register select signals are applied to RS_0 and RS_1.

Interrupt Flags (CRA-6, CRA-7, CRB-6, and CRB-7) — The four interrupt flag bits are set by active transitions of signals on the four interrupt and peripheral control lines when those lines are programmed to be inputs. These bits cannot be set directly from the MPU data bus and are reset indirectly by a read peripheral data operation on the appropriate section.

Control of CA_1 and CB_1 Interrupt Input Lines (CRA-0, CRB-0, CRA-1, and CRB-1) — The two lowest order bits of the control registers are used to control the interrupt input lines CA_1 and CB_1. Bits CRA-0 and CRB-0 are used to enable the MPU interrupt signals IRQA and IRQB, respectively. Bits CRA-1 and CRB-1 determine the active transition of the interrupt input signals CA_1 and CB_1 (see *Table 3*).

Control of CA_2 and CB_2 Peripheral Control Lines (CRA-3, CRA-4, CRA-5, CRB-3, CRB-4 and CRB-5) — Bits 3, 4 and 5 of the two control registers are used to control the CA_2 and CB_2 peripheral control lines. These bits determine if the control lines will be an interrupt input or an output control signal. If bit CRA-5 (CRB-5) is LOW, CA_2 (CB_2) is an interrupt input line similar to CA_1 (CB_1) (see *Table 4*). When CRA-5 (CRB-5) is HIGH, CA_2 (CB_2) becomes an output signal that may be used to control peripheral data transfers. When in the output mode, CA_2 and CB_2 have slightly different characteristics (see *Tables 5* and *6*).

Table 3 Control of Interrupt Inputs CA_1 and CB_1

CRA-1 (CRB-1)	CRA-0 (CRB-0)	Interrupt Input CA_1 (CB_1)	Interrupt Flag CRA-7 (CRB-7)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	↓ Active	Set HIGH on ↓ of CA_1 (CB_1)	Disabled — $\overline{IRQA}$ ($\overline{IRQB}$) remains HIGH
0	1	↓ Active	Set HIGH on ↓ of CA_1 (CB_1)	Goes LOW when the interrupt flag bit CRA-7 (CRB-7) goes HIGH
1	0	↑ Active	Set HIGH on ↑ of CA_1 (CB_1)	Disabled — $\overline{IRQA}$ ($\overline{IRQB}$) remains HIGH
1	1	↑ Active	Set HIGH on ↑ of CA_1 (CB_1)	Goes LOW when the interrupt flag bit CRA-7 (CRB-7) goes HIGH

Notes

1. ↑ indicates positive transition (LOW-to-HIGH)
2. ↓ indicates negative transition (HIGH-to-LOW)
3. The interrupt flag bit CRA-7 is cleared by an MPU read of the A data register, and CRB-7 is cleared by an MPU read of the B data register.
4. If CRA-0 (CRB-0) is LOW when an interrupt occurs (interrupt disabled) and is later brought HIGH, $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA-0 (CRB-0) is written to a "1".

Table 4 Control of CA_2 and CB_2 as Interrupt Inputs

CRA-5 (CRB-5)	CRA-4 (CRB-4)	CRA-3 (CRB-3)	Interrupt Input CA_2 (CB_2)	Interrupt Flag CRA-6 (CRB-6)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	0	↓ Active	Set HIGH on ↓ of CA_2 (CB_2)	Disabled — $\overline{IRQA}$ ($\overline{IRQB}$) remains HIGH
0	0	1	↓ Active	Set HIGH on ↓ of CA_2 (CB_2)	Goes LOW when the interrupt flag bit CRA-6 (CRB-6) goes HIGH
0	1	0	↑ Active	Set HIGH on ↑ of CA_2 (CB_2)	Disabled — $\overline{IRQA}$ ($\overline{IRQB}$) remains HIGH
0	1	1	↑ Active	Set HIGH on ↑ of CA_2 (CB_2)	Goes LOW when the interrupt flag bit CRA-6 (CRB-6) goes HIGH

Notes

1. ↑ indicates positive transition (LOW-to-HIGH)
2. ↓ indicates negative transition (HIGH-to-LOW)
3. The interrupt flag bit CRA-6 is cleared by an MPU read of the A data register, and CRB-6 is cleared by an MPU read of the B data register.
4. If CRA-3 (CRB-3) is LOW when an interrupt occurs (interrupt disabled) and is later brought HIGH, $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA-3 (CRB-3) is written to a "1".

Table 5 Control of CB_2 as an Output

CRB-5	CRB-4	CRB-3	CB_2	
			Cleared	Set
1	0	0	LOW on the positive transition of the first E pulse following an MPU write "B" data register operation.	HIGH when the interrupt flag bit CRB-7 is set by an active transition of the CB_1 signal.
1	0	1	LOW on the positive transition of the first E pulse after an MPU write "B" data register operation.	HIGH on the positive edge of the first E pulse following an E pulse that occurred while the part was deselected.
1	1	0	LOW when CRB-3 goes LOW as a result of an MPU write in control register "B".	Always LOW as long as CRB-3 is LOW. Goes HIGH on an MPU write in control register "B" that changes CRB-3 to "1".
1	1	1	Always HIGH as long as CRB-3 is HIGH. Cleared when an MPU write control register "B" results in clearing CRB-3 to "0".	HIGH when CRB-3 goes HIGH as a result of an MPU write into control register "B".

Table 6 Control of CA_2 as an Output

CRA-5	CRA-4	CRA-3	CA_2	
			Cleared	Set
1	0	0	LOW on negative transition of E after an MPU read "A" data operation.	HIGH when the interrupt flag bit CRA-7 is set by an active transition of the CA_1 signal.
1	0	1	LOW on negative transition of E after an MPU read "A" data operation.	HIGH on the negative edge of the first E pulse that occurs during a deselect.
1	1	0	LOW when CRA-3 goes LOW as a result of an MPU write to control register "A".	Always LOW as long as CRA-3 is LOW. Goes HIGH on an MPU write to control register "A" that changes CRA-3 to "1".
1	1	1	Always HIGH as long as CRA-3 is HIGH. Cleared on an MPU write to control register "A" that clears CRA-3 to a "0".	HIGH when CRA-3 goes HIGH as a result of an MPU write to control register "A".

Absolute Maximum Ratings

Supply Voltage	−0.3 V, +7.0 V
Input Voltage	−0.3 V, +7.0 V
Operating Temperature Range	0°C, +70°C
Storage Temperature Range	−55°C, +150°C
Thermal Resistance	82.5°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this hig impedance circuit.

DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = 0 to 70°C unless otherwise noted.

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
V_{IH}	Input HIGH Voltage Enable Other Inputs	 V_{SS} +2.4 V_{SS} +2.0		 V_{CC} V_{CC}	V	
V_{IL}	Input LOW Voltage Enable Other Inputs	 V_{SS} −0.3 V_{SS} −0.3		 V_{SS} +0.4 V_{SS} +0.8	V	
I_{IN}	Input Leakage Current R/$\overline{W}$, $\overline{RESET}$, RS_0, RS_1, CS_0, CS_1, $\overline{CS_2}$, CA_1, CB_1, Enable		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	3-State (OFF State) Input Current D_0-D_7, PB_0-PB_7, CB_2		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
I_{IH}	Input HIGH Current PA_0-PA_7, CA_2	−100	−250		μA	V_{IH} = 2.4 V
I_{IL}	Input LOW Current PA_0-PA_7, CA_2		−1.0	−1.6	mA	V_{IL} = 0.4 V
V_{OH}	Output HIGH Voltage D_0-D_7	V_{SS} +2.4			V	I_{Load} = −205 μA, Enable pulse width <25 μs
	Other Outputs	V_{SS} +2.4			V	I_{Load} = −100 μA, Enable pulse width <25 μs
V_{OL}	Output LOW Voltage			V_{SS} +0.4	V	I_{Load} = 1.6 mA, Enable pulse width <25 μs
I_{OH}	Output HIGH Current (Sourcing) D_0-D_7 Other Outputs	 −205 −100			 μA μA	V_{OH} = 2.4 V
	PB_0-PB_7, CB_2	−1.0	−2.5	−10	mA	V_O = 1.5 V, the current for driving other than TTL, e.g., Darling base
I_{OL}	Output LOW Current (Sinking)	1.6			mA	V_{OL} = 0.4 V
I_{LOH}	Output Leakage Current (OFF State) $\overline{IRQA}$, $\overline{IRQB}$		1.0	10	μA	V_{OH} = 2.4 V
P_D	Power Dissipation			650	mW	
C_{IN}	Input Capacitance Enable D_0-D_7 PA_0-PA_7, PB_0-PB_7, CA_2, CB_2 R/$\overline{W}$, $\overline{RESET}$, RS_0, RS_1, CS_0, CS_1, $\overline{CS_2}$, CA_1, CB_1			 20 12.5 10 7.5	pF	 V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance $\overline{IRQA}$, $\overline{IRQB}$ PB_0-PB_7			 5.0 10	pF	 V_{IN} = 0, T_A = 25°C f = 1.0 MHz

AC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = 0 to 70°C unless otherwise noted.

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
t_{PDSU}	Peripheral Data Set-up Time	200			ns	*Figure 1*
t_{CA2}	Delay Time, Enable Negative Transition to CA_2 Negative Transition			1.0	μs	*Figures 2, 3*
t_{RS1}	Delay Time, Enable Negative Transition to CA_2 Positive Transition			1.0	μs	*Figure 2*
t_r, t_f	Rise and Fall Times for CA_1 and CA_2 Input Signals			1.0	μs	*Figure 3*
t_{RS2}	Delay Time from CA_1 Active Transition to CA_2 Positive Transition			2.0	μs	*Figure 3*
t_{PDW}	Delay Time, Enable Negative Transition to Peripheral Data Valid			1.0	μs	*Figures 4, 5*
t_{CMOS}	Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid PA_0-PA_7, CA_2			2.0	μs	V_{CC} − 30% V_{CC}, *Figure 4;* *Figure 12,* Load C
t_{CB2}	Delay Time, Enable Positive Transition to CB_2 Negative Transition			1.0	μs	*Figures 6, 7*
t_{DC}	Delay Time, Peripheral Data Valid to CB_2 Negative Transition	20			ns	*Figure 5*
t_{RS1}	Delay Time, Enable Positive Transition to CB_2 Positive Transition			1.0	μs	*Figure 6*
t_r, t_f	Rise and Fall Time for CB_1 and CB_2 Input Signals			1.0	μs	*Figure 7*
t_{RS2}	Delay Time, CB_1 Active Transition to CB_2 Positive Transition			2.0	μs	*Figure 7*
t_{IR}	Interrupt Release Time, $\overline{IRQA}$ and $\overline{IRQB}$			1.6	μs	*Figure 8*
t_{RL}	$\overline{RESET}$ LOW Time	2.0			μs	*Figure 9,* Note 1

Note 1. The $\overline{RESET}$ line must be HIGH a minimum of 1.0 μs before addressing the PIA.

5

Bus Timing Characteristics

Read *Figures 10* and *12*

Symbol	Characteristic	Min	Typ	Max	Unit
t_{cycE}	Enable Cycle Time	1.0			μs
PW_{EH}	Enable Pulse Width, HIGH	0.45		25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43			μs
t_{AS}	Set-up Time, Address and R/$\overline{W}$ Valid to Enable Positive Transition	160			ns
t_{DDR}	Data Delay Time			320	ns
t_H	Data Hold Time	10			ns
t_{AH}	Address Hold Time	10			ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input			25	ns
Write *Figures 11* and *12*					
t_{cycE}	Enable Cycle Time	1.0			μs
PW_{EH}	Enable Pulse Width, HIGH	0.45		25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43			μs
t_{AS}	Set-up Time, Address and R/$\overline{W}$ Valid to Enable Positive Transition	160			ns
t_{DSW}	Data Set-up Time	195			ns
t_H	Data Hold Time	10			ns
t_{AH}	Address Hold Time	10			ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input			25	ns

Fig. 1 Peripheral Data Set-up Time (Read Mode)

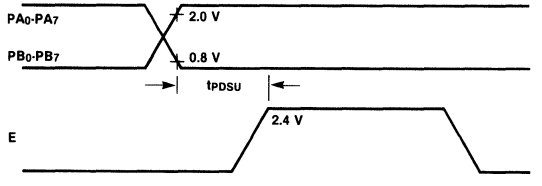
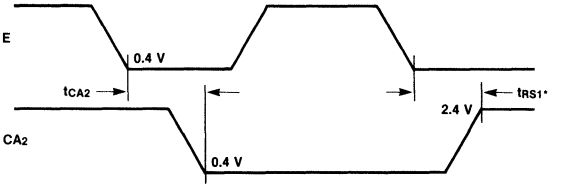


Fig. 2 CA_2 Delay Time (Read Mode; CRA-5 = CRA-3 = 1, CRA-4 = 0)



*Assumes part was deselected during any previous E pulse.

Fig. 3 CA_2 Delay Time (Read Mode; CRA-5 = 1, CRA-3 = CRA-4 = 0)

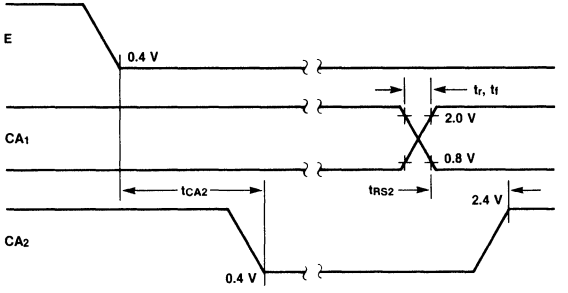


Fig. 4 Peripheral CMOS Data Delay Times (Write Mode; CRA-5 = CRA-3 = 1, CRA-4 = 0)

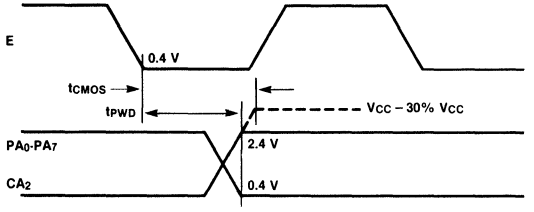
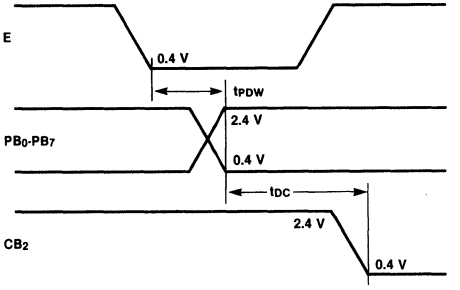
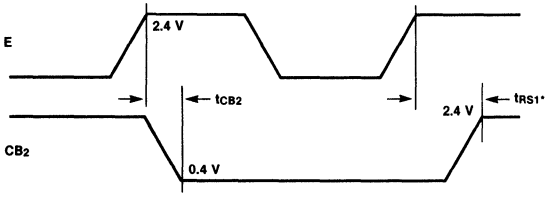


Fig. 5 Peripheral Data and CB_2 Delay Times (Write Mode; CRB-5 = CRB-3 = 1, CRB-4 = 0)



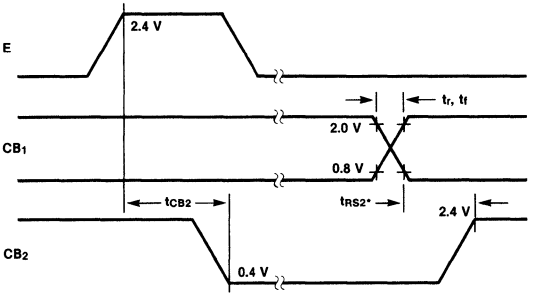
Note CB_2 goes LOW as a result of the positive transition of the E pulse.

Fig. 6 CB_2 Delay Time (Write Mode; CRB-5 = CRB-3 = 1, CRB-4 = 0)



*Assumes part was deselected during the previous E pulse.

Fig. 7 CB_2 Delay Time (Write Mode; CRB-5 = 1, CRB-3 = CRB-4 = 0)



*Assumes part was deselected during the previous E pulse.

Fig. 8 Interrupt Release Time

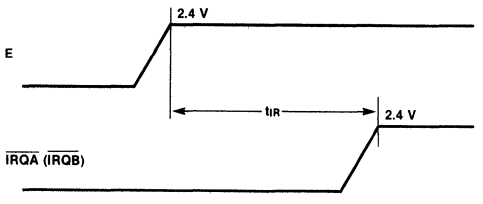
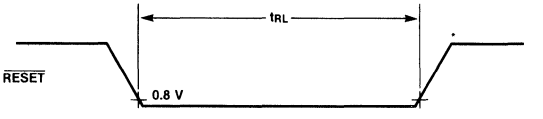


Fig. 9 $\overline{\text{RESET}}$ LOW Time



*The $\overline{\text{RESET}}$ line must be at V_{IH} for a minimum of 1.0 μs before addressing the PIA.

Fig. 10 Bus Read Timing Characteristics (Read Information from PIA)

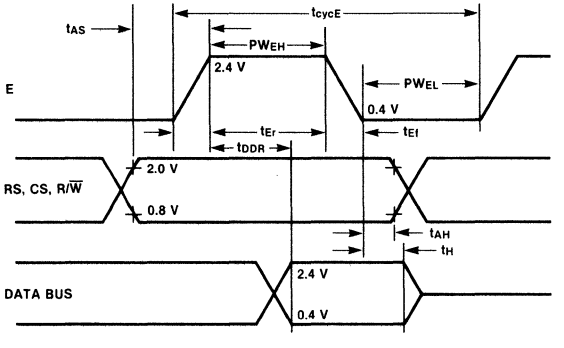


Fig. 11 Bus Write Timing Characteristics (Write Information into PIA)

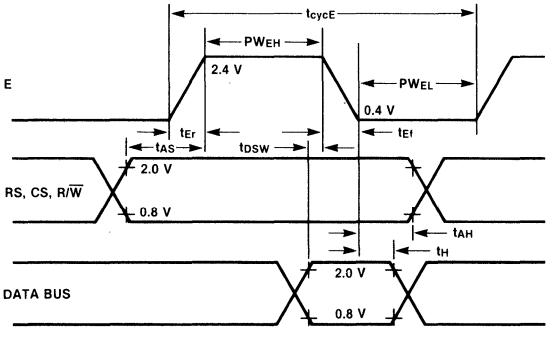
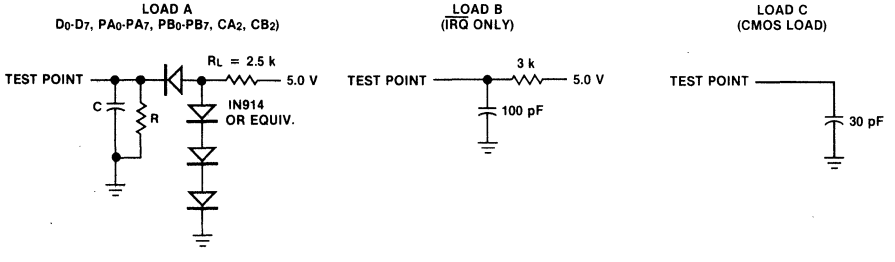


Fig. 12 Bus Timing Test Loads



C = 130 pF for D_0-D_7
= 30 pF for PA_0-PA_7, PB_0-PB_7, CA_2, and CB_2
R = 11.7 kΩ for D_0-D_7
= 24 kΩ for PA_0-PA_7, PB_0-PB_7, CA_2 and CB_2

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6820P,S	0 °C to + 70 °C

P = Plastic DIP S = Ceramic DIP

FAIRCHILD
A Schlumberger Company

F6821/F68A21/F68B21 Peripheral Interface Adapter (PIA)

Microprocessor Product

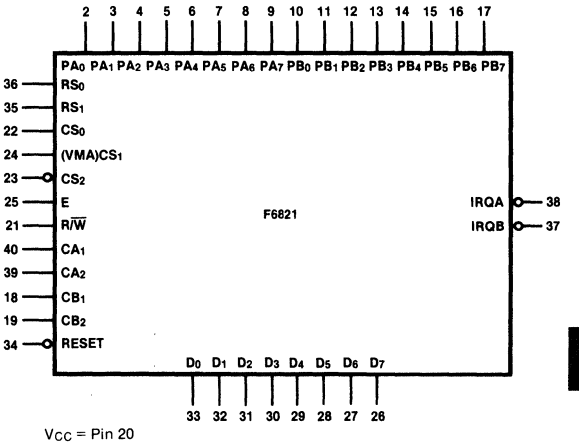
Description

The F6821 Peripheral Interface Adapter (PIA) provides a universal means of interfacing peripheral equipment to the F6800 microprocessing unit (MPU). This device is capable of interfacing the MPU to peripherals through two 8-bit bidirectional data buses and four control lines, in three speed ranges: 1.0 MHz (F6821), 1.5 MHz (F68A21), and 2.0 MHz (F68B21). No external logic is required for interfacing to most peripheral devices.

The functional configuration of the PIA is programmed by the MPU during system initialization. Each of the peripheral data lines can be programmed to act as an input or output, and each of the four control/interrupt lines may be programmed for one of several control modes. This allows a high degree of flexibility in the overall operation of the interface.

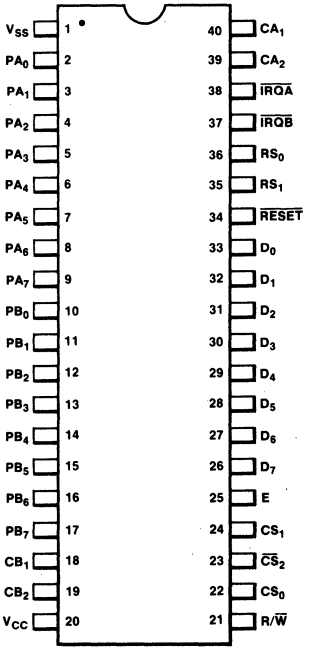
- **8-Bit Bidirectional Data Bus for Communication with the MPU**
- **Two Bidirectional 8-Bit Buses for Interface to Peripherals**
- **Two Programmable Control Registers**
- **Two Programmable Data Direction Registers**
- **Four Individually Controlled Interrupt Input Lines, Two Usable as Peripheral Control Outputs**
- **Handshake Control Logic for Input and Output Peripheral Operation**
- **High-Impedance 3-State and Direct Transistor Drive Peripheral Lines**
- **Program-Controlled Interrupt and Interrupt Disable Capability**
- **CMOS Drive Capability on Section A Peripheral Lines**
- **Two-TTL Drive Capability on All A- and B-Section Buffers**
- **TTL-Compatible**
- **Static Operation**

Logic Symbol



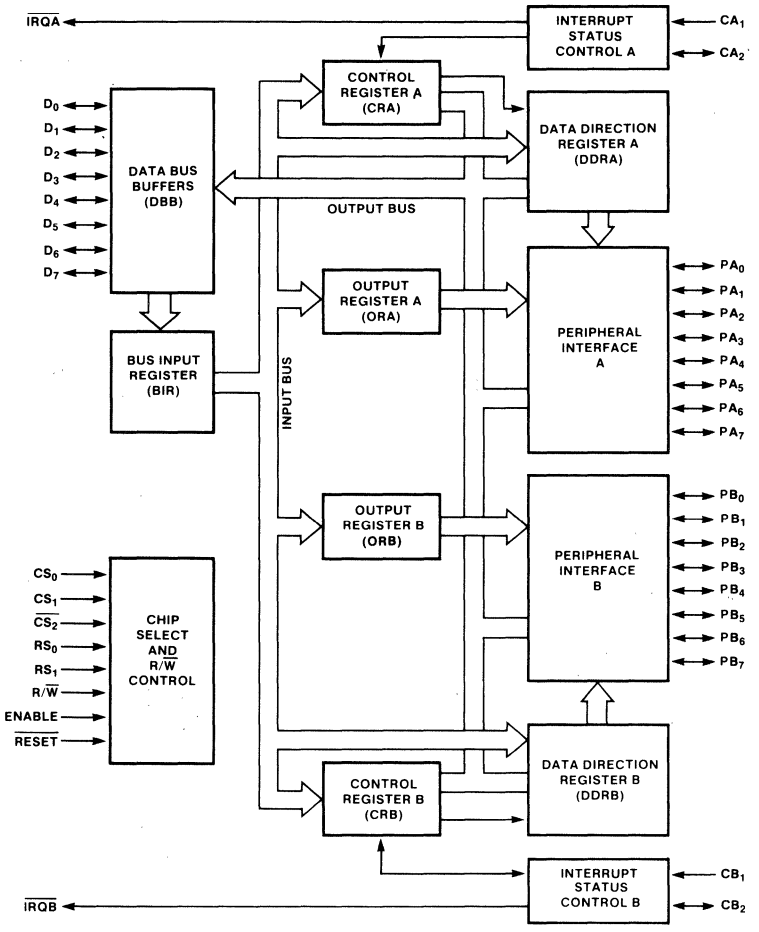
VCC = Pin 20
VSS = Pin 1

Connection Diagram
40-Pin DIP



(Top View)

Block Diagram



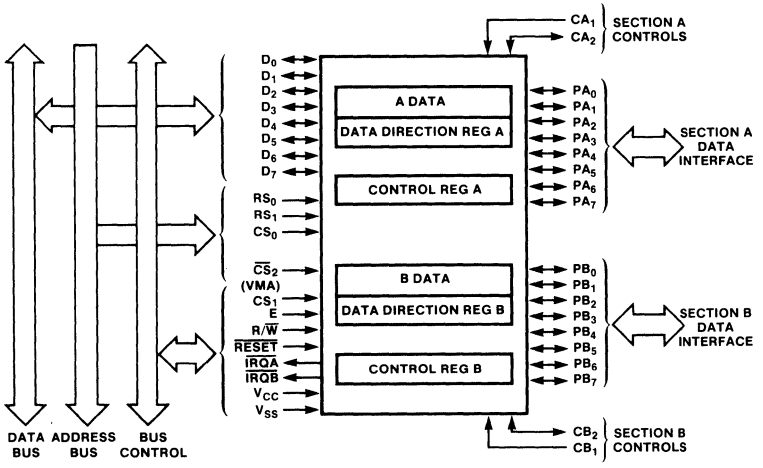
PIA/MPU Interface Signals

The PIA interfaces to the F6800 MPU with an 8-bit bidirectional data bus, three chip select lines, two register select lines, two interrupt request lines, a read/write line, an enable line, and a reset line (see *Figure 1*). These signals, in conjunction with the F6800 VMA output, permit the MPU to have complete control over the PIA. The VMA output should be utilized in conjunction with an MPU address line into a chip select of the PIA.

Data Bus (D_0 - D_7), Pins 26-33

The bidirectional data lines allow the transfer of data between the MPU and the PIA. The data bus output drivers are 3-state devices that remain in the high-impedance (OFF) state, except when the MPU performs a PIA read operation. The read/write (R/$\overline{W}$) line is in the read (HIGH) state when the PIA is selected for a read operation.

Fig. 1 PIA Bus Interface



Enable (E), Pin 25
The enable input pulse is the only timing signal that is supplied to the PIA. Timing of all other signals is referenced to the leading and trailing edges of the E pulse. This signal is normally a derivative of the $\phi 2$ F6800 clock.

Read/Write (R/$\overline{W}$), Pin 21
This input signal is generated by the MPU to control the direction of data transfer on the data bus. A LOW on the R/$\overline{W}$ line enables the input buffers and allows data transfer from the MPU to the PIA on the E signal if the device has been selected. A HIGH on the R/$\overline{W}$ line sets up the PIA for a transfer of data to the bus; the PIA output buffers are enabled when the proper address and the E pulse are present.

Reset ($\overline{RESET}$), Pin 34
The active-LOW $\overline{RESET}$ input is used to reset all register bits in the PIA to a logic 0 (LOW) state. This line can be used as a power-on reset and as a master reset during system operation.

Chip Select (CS_0 - $\overline{CS_2}$), Pins 22-24
These three input signals are used to select the PIA. The CS_0 and CS_1 lines must be HIGH and $\overline{CS_2}$ must be LOW for selection of the device. Data transfers are then performed under control of the enable and read/write signals. The device is "deselected" when any of the chip select lines is in the inactive state.

The chip select lines should be stable for the duration of the E pulse.

Register Select (RS_0, RS_1), Pins 35, 36
The two register select inputs are used to select the various registers within the PIA. These two lines are used in conjunction with internal control registers to select a particular register that is to be written to or read from.

The register select lines should be stable for the duration of the E pulse while in the read or write cycle.

Interrupt Request ($\overline{IRQA}$, $\overline{IRQB}$), Pins 37, 38
The active-LOW interrupt request inputs act to interrupt the MPU either directly or through interrupt priority circuitry. These lines are open drain (no load device on the chip). This permits all interrupt request lines to be tied together in a wired-OR configuration.

Each interrupt request line has two internal interrupt flag bits that can cause either line to go LOW. Each flag bit is associated with a particular peripheral interrupt line. Four interrupt enable bits are also provided in the PIA; these may be used to inhibit a particular interrupt from a peripheral device.

Servicing an interrupt by the MPU is accomplished by a software routine that, on a priority basis, sequentially reads and tests the two control registers in each PIA for interrupt flag bits that are set.

The interrupt flags are cleared (set to 0) as a result of an MPU read peripheral data operation of the corresponding data register. After being cleared, the interrupt flag bit cannot be enabled until the PIA is "deselected" during an E pulse. The E pulse is used to condition the interrupt control lines (CA_1, CA_2, CB_1, CB_2). When these

lines are used as interrupt inputs, at least one E pulse must occur from the inactive edge to the active edge of the interrupt input signal to condition the edge sense network. If the interrupt flag has been enabled and the edge sense circuit has been conditioned properly, the interrupt flag is set on the next active transition of the interrupt input pin.

PIA/Peripheral Interface Signals

The PIA provides two 8-bit bidirectional data buses and four interrupt/control lines for interfacing to peripheral devices.

Section A Peripheral Data (PA_0 - PA_7), Pins 2-9
Each of the peripheral data lines is programmed to act as an input or output. This is accomplished by setting a 1 in the corresponding data direction register (DDR) bit for those lines that are to be outputs. A 0 in a bit of the DDR causes the corresponding peripheral data line to act as an input. During an MPU read peripheral data operation, the data on peripheral lines programmed to act as inputs appears directly on the corresponding MPU data bus lines. In the input mode, the internal pull-up resistor on these lines represents a maximum of one standard TTL load.

The data in output register A (ORA) appears on the data lines that are programmed to be outputs. A logic 1 written into the register causes a HIGH on the corresponding data line, while a 0 results in a LOW. Data in ORA may be read by an MPU read peripheral data A operation when the corresponding lines are programmed as outputs. This data is read properly if the voltage on the peripheral data lines is greater than 2.0 V for a logic 1 output and less than 0.8 V for a logic 0 output. Loading the output lines in such a way that the voltage on these lines does not reach full voltage causes the data transferred into the MPU during a read operation to differ from that contained in the respective bit of output register A.

Section B Peripheral Data (PB_1 - PB_7), Pins 10-17
The peripheral data lines in the B section of the PIA can be programmed to act as either inputs or outputs in a manner similar to the A section lines. However, the output buffers driving these lines differ from those driving the A section lines, having a 3-state capability that allows them to enter a high-impedance state when the peripheral data line is used as an input. In addition, data on peripheral data lines PB_0 through PB_7 is read properly from those lines programmed as outputs even if the voltages are below 2.0 V for a HIGH. As outputs, these lines are compatible with standard TTL and may also be used as a source of up to 1 mA at 1.5 V to drive the base of a transistor switch directly.

Interrupt Input (CA_1, CB_1), Pins 18 and 40
The interrupt input lines are input-only lines that set the interrupt flags of the control registers. The active transition for these signals is also programmed by the two control registers.

Peripheral Control (CA_2, CB_2), Pins 39, 19
Peripheral control line CA_2 can be programmed to act as an interrupt input or as a peripheral control output. As an output, this line is compatible with standard TTL; as an input, the internal pull-up resistor on this line represents one standard TTL load. The function of this signal line is programmed by control register A (CRA).

Peripheral control line CB_2 may also be programmed to act as in interrupt input or peripheral control output. As an input, this line has high input impedance and is compatible with standard TTL. As an output, it is compatible with standard TTL and may also be used as a source of up to 1 mA at 1.5 V to drive the base of a transistor switch directly. This line is programmed by control register B (CRB).

It is recommended that the control lines (CA_1, CB_1, CA_2, CB_2) be held in a logic 1 state when the $\overline{RESET}$ line is active to prevent setting of corresponding interrupt flags in the control register when $\overline{RESET}$ goes to an inactive state. Subsequent to $\overline{RESET}$ going inactive, a read of the data registers may be used to clear any undesired interrupt flags.

Internal Controls

There are six locations within the PIA that are accessible to the MPU data bus: two peripheral registers, two data direction registers, and two control registers. Selection of these locations is controlled by the register select inputs, together with bit 2 in the control register, as shown in *Table 1.*

Table 1 Internal Addressing

		Control Register Bit		
RS_1	RS_0	CRA-2	CRB-2	Location Selected
0	0	1	X	Peripheral Register A
0	0	0	X	Data Direction Register A
0	1	X	X	Control Register A
1	0	X	1	Peripheral Register B
1	0	X	0	Data Direction Register B
1	1	X	X	Control Register B

X = Don't Care

Initialization
A LOW on the $\overline{RESET}$ line has the effect of zeroing all PIA registers. This sets PA_0 - PA_7, PB_0 - PB_7, CA_2, and CB_2

as inputs and disables all interrupts. The PIA must be configured during the restart program that follows the reset.

Register Operation

Possible configurations of the data direction and control registers are as follows:

Data Direction Registers (DDRA, DDRB)
The two data direction registers allow the MPU to control the direction of data through each corresponding peripheral data line. A DDR bit set to 0 configures the corresponding peripheral data line as an input; a 1 results in an output.

Control Registers (CRA, CRB)
The two control registers allow the MPU to control the operation of the four peripheral control lines (CA_1, CB_1, CA_2, and CB_2). In addition, they allow the MPU to enable the interrupt lines and monitor the status of the interrupt flags. Bits 0 through 5 of the two registers may be written to or read from by the MPU when the proper chip select and register select signals are applied. Bits 6 and 7 of the two registers are read-only and are modified by external interrupts occurring on the peripheral control lines. The format of the control words is shown in *Table 2.*

Table 2 Control Word Format

	7	6	5	4	3	2	1	0
CRA	$IRQA_1$	$IRQA_2$	CA_2 Control			DDRA Access	CA_1 Control	
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRB	$IRQB_1$	$IRQB_2$	CB_2 Control			DDRB Access	CB_1 Control	

5

Table 3 Interrupt Input Line Control Bits

CRA-1 (CRB-1)	CRA-0 (CRB-0)	Interrupt Input CA_1 (CB_1)	Interrupt Flag CRA-7 (CRB-7)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	↓ Active	Set HIGH on ↓ of CA_1 (CB_1).	Disabled; $\overline{IRQ}$ remains HIGH.
0	1	↓ Active	Set HIGH on ↓ of CA_1 (CB_1).	Goes LOW when interrupt flag bit CRA-7 (CRB-7) goes HIGH.
1	0	↑ Active	Set HIGH on ↑ of CA_1 (CB_1).	Disabled; $\overline{IRQ}$ remains HIGH.
1	1	↑ Active	Set HIGH on ↑ of CA_1 (CB_1).	Goes LOW when interrupt flag bit CRA-7 (CRB-7) goes HIGH.

Table 4 Peripheral Control Line Control Bits (CRA-5/CRB-5 LOW)

CRA-5 (CRB-5)	CRA-4 (CRB-4)	CRA-3 (CRB-3)	Interrupt Input CA_2 (CB_2)	Interrupt Flag CRA-6 (CRB-6)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	0	↓ Active	Set HIGH on ↓ CA_2 (CB_2).	Disabled; $\overline{IRQ}$ remains HIGH.
0	0	1	↓ Active	Set HIGH on ↓ of CA_2 (CB_2).	Goes LOW when interrupt flag bit CRA-6 (CRB-6) goes HIGH.
0	1	0	↑ Active	Set HIGH on ↑ of CA_2 (CB_2).	Disabled; $\overline{IRQ}$ remains HIGH.
0	1	1	↑ Active	Set HIGH on ↑ CA_2 (CB_2).	Goes LOW when interrupt flag bit CRA-6 (CRB-6) goes HIGH.

Notes
1. ↓ indicates negative transition (HIGH-to-LOW).
2. ↑ indicates positive transition (LOW-to-HIGH).
3. The interrupt flag bit, CRA-7, is cleared by an MPU read of the A data register, and CRB-7 is cleared by an MPU read of the B data register.
4. If CRA-0 (CRB-0) is LOW when an interrupt occurs (interrupt disabled) and is later brought HIGH, $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA-0 (CRB-0) is written to a 1.

Data Direction Access Control Bit (CRA-2, CRB-2)
Bit 2 in each control register allows selection of either a peripheral interface register (PIR) or the DDR when the proper register select signals are applied to RS_0 and RS_1.

Interrupt Flag Control Bits (CRA-6, CRA-7, CRB-6, CRB-7)
The four interrupt flag bits are set by active transitions of signals on the four interrupt and peripheral control lines when those lines are programmed to be input lines. These bits cannot be set directly from the MPU data bus and are reset indirectly by a read peripheral data operation on the appropriate section.

Interrupt Input Line Control Bits (CRA-0, CRA-1, CRB-0, CRB-1)
The two lowest-order bits of the control registers are used to control interrupt input lines CA_1 and CB_1. Bits CRA-0 and CRB-0 are used to enable MPU interrupt signals $\overline{IRQA}$ and $\overline{IRQB}$, respectively. Bits CRA-1 and CRB-1 determine the active transition of the interrupt input signals (see *Table 3*).

Peripheral Control Line Control Bits (CRA-3, CRA-4, CRA-5, CRB-3, CRB-4, CRB-5)
Bits 3, 4, and 5 of the two control registers are used to control the CA_2 and CB_2 peripheral control lines. These bits determine if the control lines act as interrupt inputs or as control outputs. If bit CRA-5 (CRB-5) is LOW, CA_2 (CB_2) is an interrupt input line similar to CA_1 (CB_1) (see *Table 4*). When CRA-5 (CRB-5) is HIGH, CA_2 (CB_2) becomes an output that may be used to control peripheral data transfers. When in the output mode, CA_2 and CB_2 have slightly different characteristics (see *Table 5* and *Table 6*).

Table 5 Control of CA_2 as an Output

CRA-5	CRA-4	CRA-3	CA_2	
			Cleared	Set
1	0	0	LOW on the negative transition of E after an MPU read data register A operation.	HIGH when interrupt flag bit CRA-7 is set by an active transition of the CA_1 signal.
1	0	1	LOW on the negative transition of E after an MPU read data register A operation.	HIGH on the negative edge of the first E pulse that occurs while the device is deselected.
1	1	0	LOW when CRA-3 goes LOW as a result of an MPU write control register A operation.	Always LOW as long as CRA-3 is LOW. Goes HIGH on an MPU write control register A operation that changes CRA-3 to 1.
1	1	1	Always HIGH as long as CRA-3 is HIGH. Cleared on an MPU write control register A operation that clears CRA-3 to 0.	HIGH when CRA-3 goes HIGH as a result of an MPU write control register A operation.

Table 6 Control of CB_2 as an Output

CRB-5	CRB-4	CRB-3	CB_2	
			Cleared	Set
1	0	0	LOW on the positive transition of the first E pulse following an MPU write data register B operation.	HIGH when interrupt flag bit CRB-7 is set by an active transition of the CB_1 signal.
1	0	1	LOW on the positive transition of the first E pulse after an MPU write data register B operation.	HIGH on the positive edge of the first E pulse following an E pulse that occurred while the device was deselected.
1	1	0	LOW when CRB-3 goes LOW as a result of an MPU write control register B operation.	Always LOW as long as CRB-3 is LOW. Goes HIGH on an MPU write control register B operation that changes CRB-3 to 1.
1	1	1	Always HIGH as long as CRB-3 is HIGH. Cleared when an MPU write control register B operation results in clearing CRB-3 to 0.	HIGH when CRB-3 goes HIGH as a result of an MPU write control register B operation.

Absolute Maximum Ratings

Supply Voltage	-0.3 V, +7.0 V
Input Voltage	-0.3 V, +7.0 V
Operating Temperature — T_L to T_H	
F6821, F68A21, F68B21	0° C, +70° C
F6821C, F68A21C	-40° C, +85° C
F6821DM	-55° C, +125° C
Storage Temperature Range	-55° C, +150° C
Thermal Resistance	82.5° C/W

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

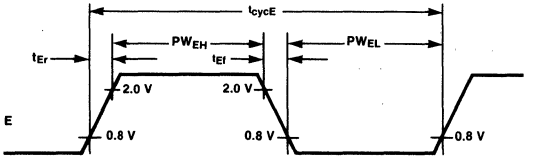
DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted.

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
Bus Control Inputs (R/$\overline{W}$, $\overline{RESET}$, RS_0, RS_1, CS_0, CS_1, $\overline{CS_2}$)						
V_{IH}	Input HIGH Voltage	V_{SS} + 2.0		V_{CC}	V	
V_{IL}	Input LOW Voltage	V_{SS} - 0.3		V_{SS} + 0.8	V	
I_{IN}	Input Leakage Current		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
C_{IN}	Capacitance			7.5	pF	V_{IN} = 0, T_A = 25° C, f = 1.0 MHz
Interrupt Outputs ($\overline{IRQA}$, $\overline{IRQB}$)						
V_{OL}	Output LOW Voltage			V_{SS} + 0.4	V	I_{Load} = 3.2 mA
I_{LOH}	Output Leakage Current (OFF-State)		1.0	10	μA	V_{OH} = 2.4 V
C_{OUT}	Capacitance			5.0	pF	V_{IN} = 0, T_A = 25° C, f = 1.0 MHz
Data Bus (D_0-D_7)						
V_{IH}	Input HIGH Voltage	V_{SS} + 2.0		V_{CC}	V	
V_{IL}	Input LOW Voltage	V_{SS} - 0.3		V_{SS} + 0.8	V	
I_{TSI}	3-State (OFF-State) Input Current		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
V_{OH}	Output HIGH Voltage	V_{SS} + 2.4			V	I_{LOAD} = -205 μA
V_{OL}	Output LOW Voltage			V_{SS} + 0.4	V	I_{LOAD} = 1.6 mA
C_{IN}	Capacitance			12.5	pF	V_{IN} = 0, T_A = 25° C, f = 1.0 MHz
Peripheral Bus (PA_0-PA_7, PB_0-PB_7, CA_1, CA_2, CB_1, CB_2)						
I_{IN}	Input Leakage Current CA_1, CB_1		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	3-State (OFF-State) Input Current PB_0-PB_7, CB_2		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
I_{IH}	Input HIGH Current PA_0-PA_7, CA_2	-200	-400		μA	V_{IH} = 2.4 V
I_{OH}	Darlington Dr. Curr. PB_0-PB_7, CB_2	-1.0		-10	mA	V_O = 1.5 V
I_{IL}	Input LOW Current PA_0-PA_7, CA_2		-1.3	-2.4	mA	V_{IL} = 0.4 V
V_{OH}	Output HIGH Voltage PA_0-PA_7, PB_0-PB_7, CA_2, CB_2 PA_0-PA_7, CA_2	 V_{SS} + 2.4 V_{CC} - 1.0			V	 I_{Load} = -200 μA I_{Load} = -10 μA
V_{OL}	Output LOW Voltage			V_{SS} + 0.4	V	I_{Load} = 3.2 mA
C_{IN}	Capacitance			10	pF	V_{IN} = 0, T_A = 25° C, f = 1.0 MHz
Power Requirements						
P_D	Power Dissipation			550	mW	

Enable Signal Timing Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted.

Symbol	Characteristic	F6821		F68A21		F68B21		Unit	Figure
		Min	Max	Min	Max	Min	Max		
t_{cycE}	Enable Cycle Time	1000		666		500		ns	2
PW_{EH}	Enable Pulse Width, HIGH	450		280		220		ns	2
PW_{EL}	Enable Pulse Width, LOW	430		280		210		ns	2
t_{Er}, t_{Ef}	Enable Pulse Rise and Fall Times		25		25		25	ns	2

Fig. 2 Enable Signal Timing Characteristics



Bus Timing Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted.

Symbol	Characteristic	F6821		F68A21		F68B21		Unit	Figures
		Min	Max	Min	Max	Min	Max		
t_{AS}	Set-Up Time, Address and R/$\overline{W}$ Valid to Enable Positive Transition	160		140		70		ns	3, 4
t_{AH}	Address Hold Time	10		10		10		ns	3, 4
t_{DDR}	Data Delay Time, Read		320		220		180	ns	3, 5
t_{DHR}	Data Hold Time, Read	10		10		10		ns	3, 5
t_{DSW}	Data Set-Up Time, Write	195		80		60		ns	4, 5
t_{DHW}	Data Hold Time, Write	10		10		10		ns	4, 5

Fig. 3 Bus Timing Characteristics (Read from PIA)

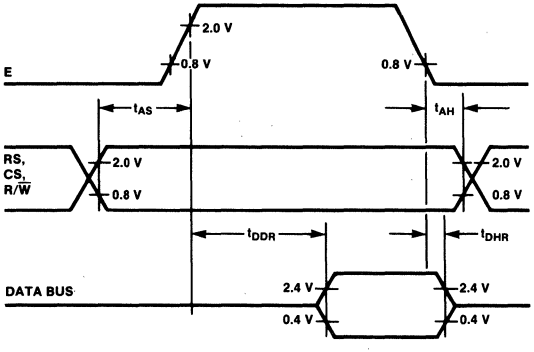
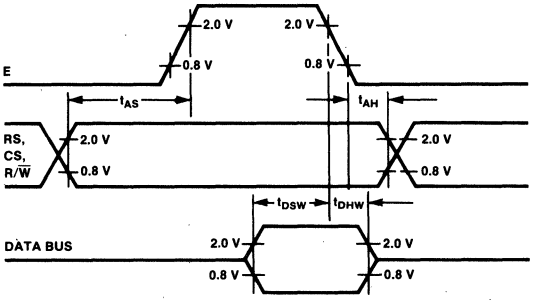


Fig. 4 Bus Timing Characteristics (Write to PIA)



Peripheral Timing Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted.

Symbol	Characteristic	F6821		F68A21		F68B21		Unit	Figures
		Min	Max	Min	Max	Min	Max		
t_{PDSU}	Peripheral Data Set-Up Time	200		135		100		ns	9
t_{PDH}	Peripheral Data Hold Time	0		0		0		ns	9
t_{CA_2}	Delay Time, Enable Negative Transition to CA_2 Negative Transition		1.0		0.670		0.500	μs	6, 10, 11
t_{RS_1}	Delay Time, Enable Negative Transition to CA_2 Positive Transition		1.0		0.670		0.500	μs	6, 10
t_r, t_f	Rise and Fall for CA_1 and CA_2 Input Signals		1.0		1.0		1.0	μs	6, 11
t_{RS_2}	Delay Time from CA_1 Active Transition to CA_2 Positive Transition		2.0		1.35		1.0	μs	6, 11
t_{PDW}	Delay Time, Enable Negative Transition to Peripheral Data Valid		1.0		0.670		0.5	μs	6, 12, 13
t_{CMOS}	Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid PA_0-PA_7, CA_2		2.0		1.35		1.0	μs	7, 12
t_{CB_2}	Delay Time, Enable Positive Transition to CB_2 Negative Transition		1.0		0.670		0.5	μs	6, 14, 15
t_{DC}	Delay Time, Peripheral Data Valid to CB_2 Negative Transition	20		20		20		ns	6, 13
t_{RS_1}	Delay Time, Enable Positive Transition to CB_2 Positive Transition		1.0		0.670		0.5	μs	6, 14
PW_{CT}	Peripheral Control Output Pulse Width, CA_2/CB_2	550		550		550		ns	6, 10, 14
t_r, t_f	Rise and Fall Time for CB_1 and CB_2 Input Signals		1.0		1.0		1.0	μs	15
t_{RS_2}	Delay Time, CB_1 Active Transition to CB_2 Positive Transition		2.0		1.35		1.0	μs	6, 15
t_{IR}	Interrupt Release Time, $\overline{IRQA}$ and $\overline{IRQB}$		1.60		1.10		0.85	μs	8, 17
t_{RS_3}	Interrupt Response Time		1.0		1.0		1.0	μs	8, 16
PW_I	Interrupt Input Pulse Width	500		500		500		ns	16
t_{RL}	Reset LOW Time*	1.0		0.66		0.5		μs	18

*The $\overline{RESET}$ line must be HIGH a minimum of 1.0 μs before addressing the PIA.

5

Fig. 5 Bus Timing Test Load

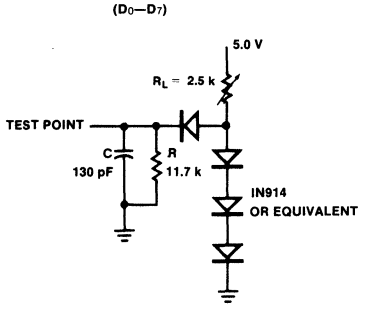
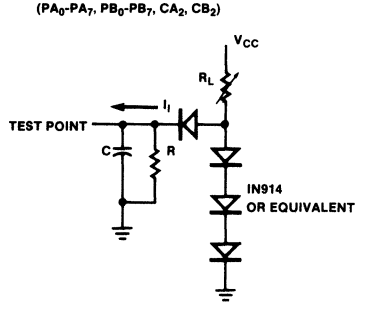


Fig. 6 TTL Equivalent Test Load



C = 40 pF, R = 12 k
Adjust R_L so that I_I = 3.2 mA
with V_I = 0.4 V and V_{CC} = 5.25 V

Fig. 7 CMOS Equivalent Test Load

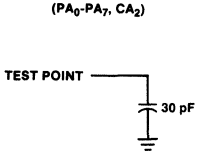


Fig. 8 NMOS Equivalent Test Load

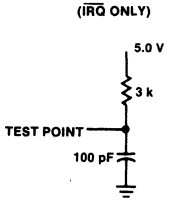


Fig. 9 Peripheral Data Set-Up and Hold Times (Read Mode)

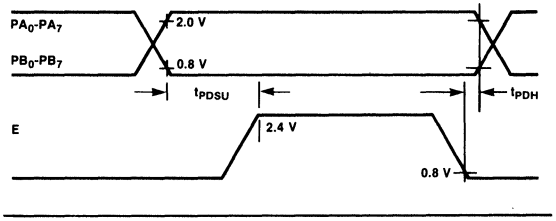
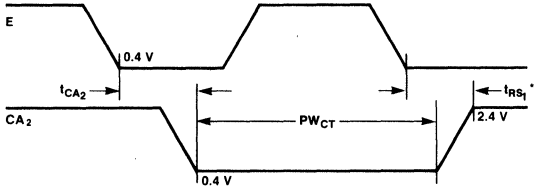


Fig. 10 CA_2 Delay Time (Read Mode; CRA-5 = CRA-3 = 1; CRA-4 = 0)



*Assumes part was deselected during the previous E pulse.

Fig. 11 CA_2 Delay Time (Read Mode; CRA-5 = 1; CRA-3 = CRA-4 = 0)

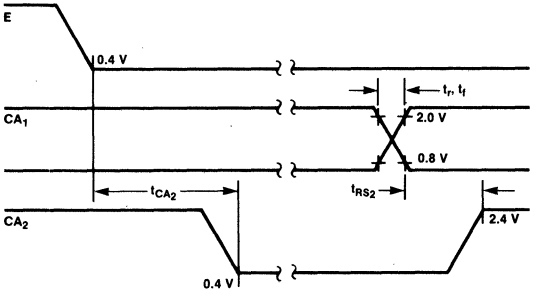


Fig. 12 Peripheral CMOS Data Delay Times (Write Mode; CRA-5 = CRA-3 = 1; CRA-4 = 0)

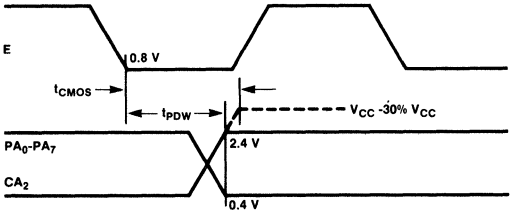
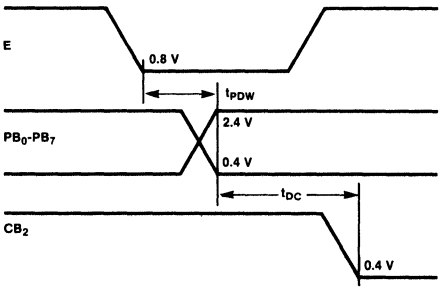
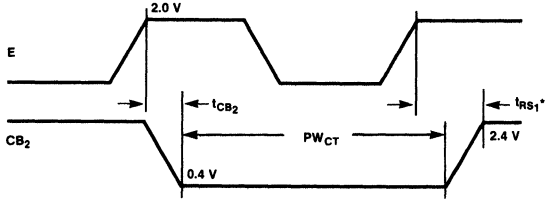


Fig. 13 Peripheral Data and CB_2 Delay Times (Write Mode; CRB-5 = CRB-3 = 1; CRB-4 = 0)



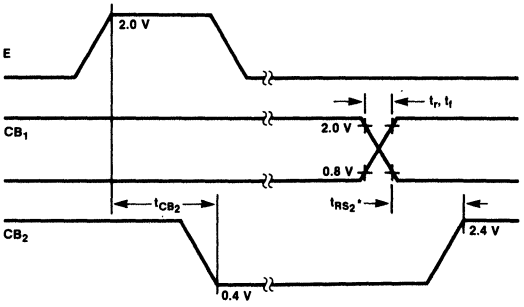
Note: CB_2 goes LOW as a result of the positive transition of the E pulse.

Fig. 14 CB_2 Delay Time (Write Mode; CRB-5 = CRB-3 = 1; CRB-4 = 0)



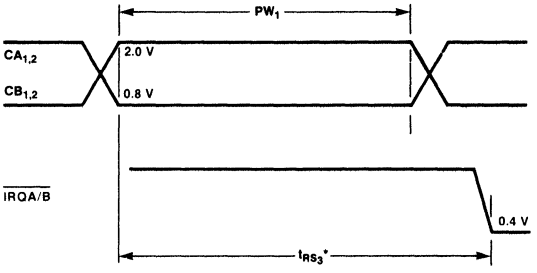
*Assumes part was deselected during the previous E pulse.

Fig. 15 CB_2 Delay Time (Write Mode; CRB-5 = 1; CRB-3 = CRB-4 = 0)



*Assumes part was deselected during any previous E pulse.

Fig. 16 Interrupt Pulse Width and $\overline{IRQ}$ Response



*Assumes interrupt enable bits are set.

Fig. 17 $\overline{IRQ}$ Release Time

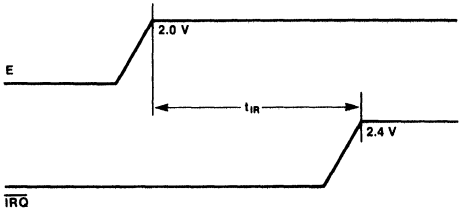
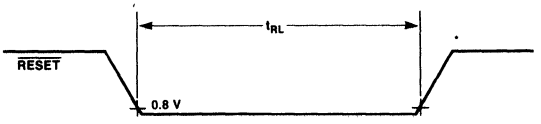


Fig. 18 $\overline{RESET}$ LOW Time



*The $\overline{RESET}$ line must be at V_{IH} for a minimum of 1.0 μs before addressing the PIA.

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6821P, S	0°C to +70°C
	F6821CP, CS	-40°C to +85°C
	F6821DM	-55°C to +125°C
1.5 MHz	F68A21P, S	0°C to +70°C
	F68A21CP, CS	-40°C to +85°C
2.0 MHz	F68B21P, S	0°C to +70°C
	F68B21DM (Waivers)	-55°C to +125°C

P = Plastic package, S = Ceramic package

FAIRCHILD
A Schlumberger Company

F6840/F68A40/F68B40 Programmable Timer (PTM)

Microprocessor Product

Description

The F6840 is a programmable subsystem component of the F6800 family designed to provide variable system time intervals.

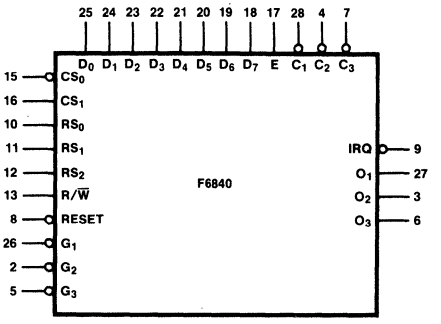
The F6840 has three 16-bit binary counters, three corresponding control registers and a status register. These counters are under software control and may be used to cause system interrupts and/or generate output signals. The F6840 may be utilized for such tasks as frequency measurements, event counting, interval measuring and similar tasks. The device may be used for square wave generation, gated delay signals, single pulses of controlled duration, and pulse width modulation, as well as system interrupts.

- **Operates From a Single +5V Power Supply**
- **Fully TTL-Compatible**
- **Single System Clock Required (Enable)**
- **Selectable Prescaler on Timer 3 Capable of 4 MHz for the F6840, 6 MHz for the F68A40, and 8 MHz for the F68B40**
- **Programmable Interrupt ($\overline{IRQ}$) Output to MPU**
- **Readable Down Counter Indicates Counts to Go to Time-Out**
- **Selectable Gating for Frequency or Pulse-Width Comparison**
- **$\overline{RESET}$ Input**
- **Three Asynchronous External Clock and Gate/Trigger Inputs Internally Synchronized**
- **Three Maskable Outputs**

Pin Names

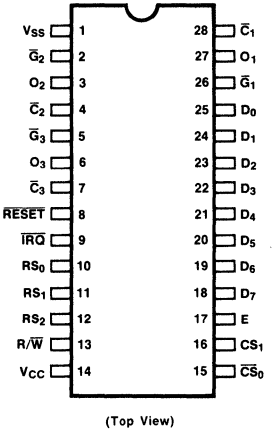
D_0-D_7	Bidirectional Data Lines
$\overline{CS}_0$-CS_1	Chip Select Input
R/$\overline{W}$	Read/Write Input
E	Enable (Systems Clock $\phi 2$) Input
$\overline{IRQ}$	Interrupt Request Output
$\overline{RESET}$	Reset Input
RS_0-RS_2	Register Select Inputs
$\overline{C}_1$-$\overline{C}_3$	Counter Clock Inputs
$\overline{G}_1$-$\overline{G}_3$	Counter Gate Inputs
O_1-O_3	Counter Outputs

Logic Symbol



V_{CC} = Pin 14
V_{SS} = Pin 1

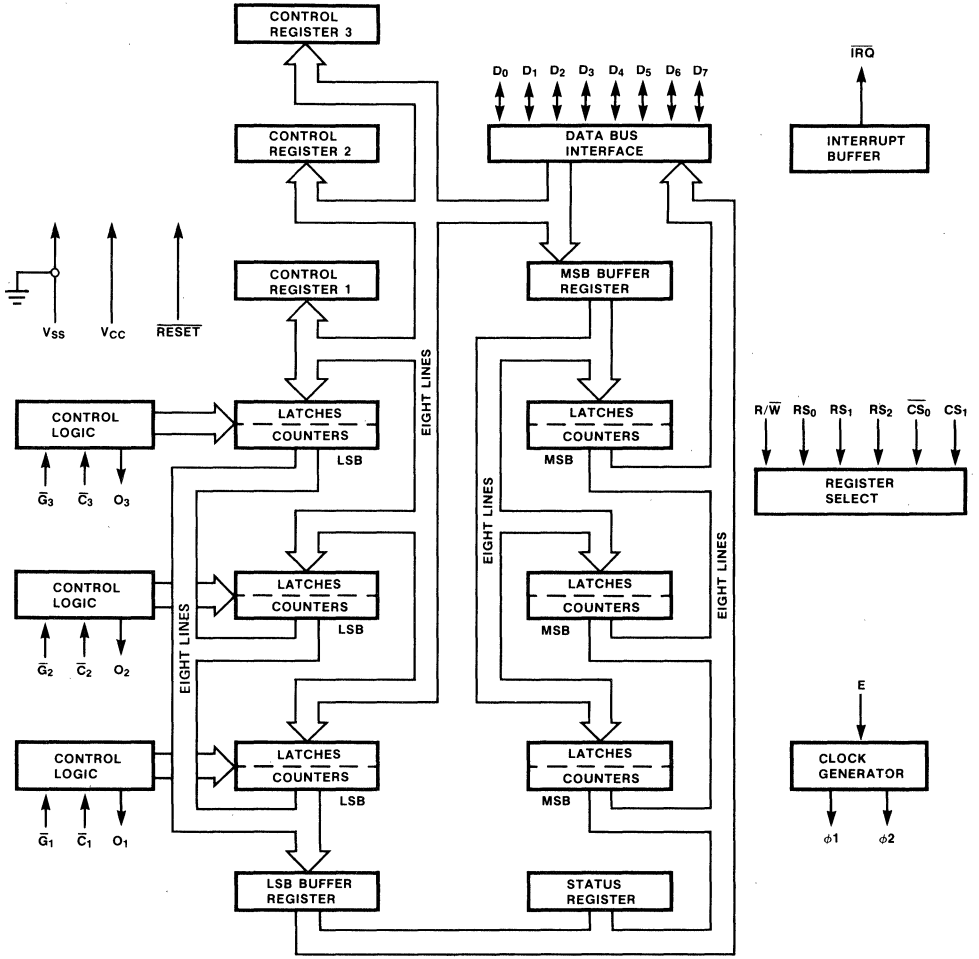
Connection Diagram
28-Pin DIP



(Top View)

5

Block Diagram



Functional Description

The three timers in the F6840 may be programmed independently to operate in modes which fit a wide variety of applications. The device is fully bus-compatible with F6800 systems and is accessed by load and store operations from the MPU in much the same manner as a memory device. In a typical application, a timer will be loaded by storing two bytes of data into an associated counter latch. This data then is transferred into the counter during a counter initialization cycle. The counter decrements on each subsequent clock period, which may be an external clock or Enable (System $\phi 2$) until one of several predetermined conditions causes it to halt or recycle. The timers are thus programmable, cyclic in nature, controllable by external inputs or the MPU program, and accessible by the MPU at any time.

Bus Interface

The programmable timer module (PTM) interfaces to the F6800 bus with an 8-bit bidirectional data bus, two Chip Select lines, a Read/Write line, an Enable (System $\phi 2$)

line, an Interrupt Request line, an external $\overline{\text{RESET}}$ line, and three Register Select lines. These signals, in conjunction with the F6800 VMA output, permit the MPU to control the PTM. VMA should be utilized in conjunction with the MPU address line into a Chip Select of the PTM.

Bidirectional Data (D_0-D_7)
The bidirectional Data Lines (D_0-D_7) allow the transfer of data between the MPU and the PTM. The data bus output drivers are 3-state devices which remain in the high-impedance (OFF) state except when the MPU performs a PTM read operation (Read/Write and Enable lines HIGH and PTM Chip Selects activated).

Chip Select ($\overline{CS_0}$, CS_1)
These two signals are used to activate the data bus interface and allow transfer of data from the PTM. With $\overline{CS_0}$ = "0" and CS_1 = "1", the device is selected and data transfer will occur.

Read/Write (R/$\overline{W}$)
This signal is generated by the MPU to control the direction of data transfer on the data bus. With the PTM selected, a LOW state on the PTM R/$\overline{W}$ line enables the input buffers and data is transferred from the MPU to the PTM on the trailing edge of the Enable (System $\phi 2$) signal. Alternately (under the same conditions), R/$\overline{W}$ = "1" and Enable HIGH allows data in the PTM to be read by the MPU.

Enable (E, System $\phi 2$)
This signal synchronizes data transfer between the MPU and the PTM. It also performs an equivalent synchronization function on the external Clock, $\overline{\text{RESET}}$, and Gate inputs of the PTM.

Interrupt Request ($\overline{IRQ}$)
The active LOW Interrupt Request signal is normally tied directly (or through priority interrupt circuitry) to the $\overline{IRQ}$ input of the MPU. This is an open drain output (no load device on the chip) which permits other similar Interrupt Request lines to be tied together in a wired-OR configuration.

The $\overline{IRQ}$ line is activated if, and only if, the composite interrupt flag (bit 7 of the internal status register) is asserted. The conditions under which the IRQ line is activated are discussed in conjunction with the status register.

External $\overline{\text{RESET}}$
A LOW level at this input is clocked into the PTM by the Enable (System $\phi 2$) input. Two Enable pulses are required to synchronize and process the signal. The PTM then recognizes the active LOW or inactive HIGH on the third Enable pulse. If the $\overline{\text{RESET}}$ signal is asynchronous, an additional Enable period is required if set-up times are not met. The $\overline{\text{RESET}}$ input must be stable HIGH/LOW for the minimum time stated in the *AC Characteristics* table.

Recognition of a LOW level at this input by the PTM causes the following action to occur:
a. All counter latches are preset to their maximal count values.
b. All control register bits are cleared with the exception of $CR1_0$ (internal reset bit), which is set.
c. All counters are preset to the contents of the latches.
d. All counter outputs are reset and all counter clocks are disabled.
e. All status register bits (interrupt flags) are cleared.

Table 1 Register Selection

Register Select Inputs			Operations	
RS_2	RS_1	RS_0	R/$\overline{W}$ = "0"	R/$\overline{W}$ = "1"
0	0	0	$CR2_0$ = "0" Write Control Register 3 $CR2_0$ = "1" Write Control Register 1	No Operation
0	0	1	Write Control Register 2	Read Status Register
0	1	0	Write MSB Buffer Register	Read Timer 1 Counter
0	1	1	Write Timer 1 Latches	Read LSB Buffer Register
1	0	0	Write MSB Buffer Register	Read Timer 2 Counter
1	0	1	Write Timer 2 Latches	Read LSB Buffer Register
1	1	0	Write MSB Buffer Register	Read Timer 3 Counter
1	1	1	Write Timer 3 Latches	Read LSB Buffer Register

Register Select Lines (RS_0, RS_1, RS_2)
These inputs are used in conjunction with the R/$\overline{W}$ line to select the internal registers, counters and latches as shown in *Table 1*.

It has been stated previously that the PTM is accessed via MPU load and store operations in much the same manner as a memory device. The instructions available with the F6800 family of MPUs which perform operations directly on memory should not be used when the PTM is accessed. These instructions actually fetch a byte from memory, perform an operation, then restore it to the same address location. Since the PTM uses the R/$\overline{W}$ line as an additional register select input, the modified data may not be restored to the same register if these instructions are used.

Control Register
Three write-only registers in the F6840 are used to modify timer operation to suit a variety of applications. Control register 2 has a unique address space (RS_0 = "1", RS_1 = "0", RS_2 = "0") and therefore may be written into any time. The remaining control registers (1 and 3) share the address space selected by a logic "0" on all register select inputs. The least significant bit of control register 2 ($CR2_0$) is used as an additional addressing bit for control registers 1 and 3. Thus, with all Register Selects and R/$\overline{W}$ inputs at logic "0", control register 3 will be written into if $CR2_0$ is a logic "0". Control register 3 can also be written into after a reset LOW condition has occurred, since all control register bits (except $CR1_0$) are cleared. Therefore, one may write in the sequence CR_3, CR_2, CR_1.

The least significant bit of control register 1 is used as an internal reset bit. When this bit is a logic "0", all timers are allowed to operate in the modes prescribed by the remaining bits of the control registers. Writing a "1" into $CR1_0$ causes all counters to be preset with the contents of the corresponding counter latches, all counter clocks to be disabled, and the timer outputs and interrupt flags (status register) to be reset. Counter latches and control registers are undisturbed by an internal reset and may be written into regardless of the state of $CR1_0$.

The least significant bit of control register 3 is used as a selector for a ÷8 prescaler, which is available with timer 3 only. The prescaler, if selected, is effectively placed between the clock input circuitry and the input to counter 3. It therefore can be used with either the internal clock (Enable) or an external clock source.

The functions depicted in the foregoing discussions are tabulated on the first row in *Table 2* for ease of reference.

Control register bits $CR1_0$, $CR2_0$ and $CR3_0$ are unique in that each selects a different function. The remaining bits (1 through 7) of each control register select common functions, with a particular control register affecting only

Table 2 Control Register Bits

$CR1_0$ Internal Reset Bit	**$CR2_0$ Control Register Address Bit**	**$CR3_0$ Timer 3 Clock Control**
0 All timers allowed to operate	**0** CR3 may be written	**0** T_3 $\overline{\text{Clock}}$ is not prescaled
1 All timers held in preset state	**1** CR1 may be written	**1** T_3 $\overline{\text{Clock}}$ is prescaled by ÷8
CRX_1*	Timer X $\overline{\text{Clock}}$ Source	
0	TX uses external clock source on $\overline{CX}$ input	
1	TX uses Enable clock	
CRX_2	Timer X Counting Mode Control	
0	TX configured for normal (16-bit) counting mode	
1	TX configured for dual 8-bit counting mode	
CRX_3 CRX_4 CRX_5	Timer X Counter Mode and Interrupt Control (See *Table 3*)	
CRX_6	Timer X Interrupt Enable	
0	Interrupt Flag masked on $\overline{IRQ}$	
1	Interrupt Flag enabled to $\overline{IRQ}$	
CRX_7	Timer X counter Output Enable	
0	TX Output masked on output OX	
1	TX Output enabled on output OX	

*Control Register for timer 1, 2 or 3, Bit 1.

its corresponding timer. For example, bit 1 of control register 1 ($CR1_1$) selects whether an internal or external clock source is to be used with timer 1. Similarly, $CR2_1$ selects the clock source for timer 2, and $CR3_1$ performs this function for timer 3. The function of each bit of control register "X" can therefore be defined as shown in the remaining section of *Table 2*.

Control register bit 2 selects whether the binary information contained in the counter latches (and subsequently loaded into the counter) is to be treated as a single 16-bit word or two 8-bit bytes. In the single 16-bit counter mode (CR2 = "0"), the counter will decrement to zero after N + 1 enabled ($\overline{G}$ = "0") clock periods, where N is defined as the 16-bit number in the counter latches. With CRX_2 = "1", a similar time-out will occur after (L + 1) · (M + 1) enabled clock periods, where L and M, respectively, refer to the LSB and MSB bytes in the counter latches.

Control register bits 3, 4, and 5 are explained in detail in the Timer Operating Modes section. Bit 6 is an interrupt mask bit which will be explained more fully in conjunction with the status register, and bit 7 is used to enable the corresponding timer output. A summary of control register programming modes is shown in *Table 3*.

Status Register/Interrupt Flags

The F6840 has an internal read-only status register which contains four interrupt flags. (The remaining four bits of the register are not used, and default to "0s" when being read). Bits 0, 1, and 2 are assigned to timers 1, 2, and 3, respectively, as individual flag bits, while bit 7 is a composite interrupt flag. This flag bit will be asserted if any of the individual flag bits is set while bit 6 of the corresponding control register is at a logic "1". The conditions for asserting the composite interrupt flag bit can therefore be expressed as:

$$INT = I_1 \cdot CR1_6 + I_2 \cdot CR2_6 + I_3 \cdot CR3_6$$

where INT = Composite Interrupt Flag (Bit 7)
I_1 = Timer 1 Interrupt Flag (Bit 0)
I_2 = Timer 2 Interrupt Flag (Bit 1)
I_3 = Timer 3 Interrupt Flag (Bit 2)

An interrupt flag is cleared by a timer reset condition; i.e., external $\overline{RESET}$ = "0" or internal reset bit ($CR1_0$) = "1". It will also be cleared by a read timer counter command, provided that the status register has previously been read while the interrupt flag was set. This condition on the read status register — read timer counter (RS-RT) sequence is designed to prevent missing interrupts which might occur after the status register is read, but prior to reading the timer counter.

An individual interrupt flag is also cleared by a write timer latches (W) command or a counter initialization (CI) sequence, provided that W or CI affects the timer corresponding to the individual interrupt flag.

Counter Latch Initialization

Each of the three independent timers consists of a 16-bit addressable counter and 16 bits of addressable latches. The counters are preset to the binary numbers stored in the latches. Counter initialization results in the transfer of the latch contents to the counter. See the notes in *Table 5* regarding the binary number N, L or M placed into the latches and their relationship to the output waveforms and counter time outs.

Since the PTM data bus is 8 bits wide and the counters are 16 bits wide, a temporary register (MSB buffer register) is provided. This write-only register is for the most significant byte of the desired latch data. Three addresses are provided for the MSB buffer register (as indicated in *Table 1*), but they all lead to the same buffer. Data from the MSB buffer will be transferred automatically into the most significant byte of timer X when a write timer X latches command is performed. So it can be seen that the F6840 has been designed to allow transfer of two bytes of data into the counter latches provided that the MSB is transferred first.

In the many applications, the source of the data will be an F6800 MPU. It should be noted that the 16-bit store operations of F6800 family microprocessors (STS and STX) transfer data in the order required by the PTM. A store index register instruction, for example, results in the MSB of the X register being transferred to the selected address, then the LSB of the X register being written into the next higher location. Thus, either the index register or stack pointer may be transferred directly into a selected counter latch with a single instruction.

A logic "0" at the $\overline{RESET}$ input also initializes the counter latches. In this case, all latches will assume a maximum count of $65{,}536_{10}$. It is important to note that an internal reset (bit zero of control register 1 set) has no effect on the counter latches.

Counter Initialization

Counter initialization is defined as the transfer of data from the latches to the counter with subsequent clearing of the individual interrupt flag associated with the counter. Counter initialization always occurs when a reset condition ($\overline{RESET}$) = "0" or $CR1_0$ = "1" is recognized. It can also occur — depending on timer mode — with a write timer latches command or recognition of a negative transition of the gate input.

Table 3 Control Register Programming

7	6	5	4	3	2	1	0	Register 1	Register 2	Register 3
X	X	X	X	X	X	X	↕	0 All timers operate	Reg #3 may be written	T3 Clk ÷ 1
								1 All timers preset	Reg #1 may be written	T3 Clk ÷ 8
X	X	X	X	X	X	↕	X	0 External Clock ($\overline{CX}$ Input)		
								1 Internal Clock (Enable)		
X	X	X	X	X	↕	X	X	0 Normal (16-Bit) Count Mode		
								1 Dual 8-Bit Count Mode		
X	X	0	0	0	X	X	X	Continuous Operating Mode: $\overline{Gate}$ ↓ or Write to Latches or Reset Causes Counter Initialization		
X	X	0	0	1	X	X	X	Frequency Comparison Mode: Interrupt if $\overline{Gate}$ ↓ ⌐ ↓ is < Counter Time-Out		
X	X	0	1	0	X	X	X	Continuous Operating Mode: $\overline{Gate}$ ↓ or Reset Causes Counter Initialization		
X	X	0	1	1	X	X	X	Pulse Width Comparison Mode: Interrupt if $\overline{Gate}$ ↓ ‾ ↑ is < Counter Time-Out		
1	X	1	0	0	X	X	X	Single Shot Mode: $\overline{Gate}$ ↓ or Write to Latches or Reset Causes Counter Initialization		
X	X	1	0	1	X	X	X	Frequency Comparison Mode: Interrupt if $\overline{Gate}$ ↓ ⌐ ↓ is > Counter Time-Out		
1	X	1	1	0	X	X	X	Single Shot Mode: $\overline{Gate}$ ↓ or Reset Causes Counter Initialization		
X	X	1	1	1	X	X	X	Pulse Width Comparison Mode: Interrupt if $\overline{Gate}$ ↓ ‾ ↑ is > Counter Time-Out		
X	↕	X	X	X	X	X	X	0 Interrupt Flag Masked ($\overline{IRQ}$)		
								1 Interrupt Flag Enabled ($\overline{IRQ}$)		
↕	X	X	X	X	X	X	X	0 Timer Output Masked		
								1 Timer Output Enabled		

Note
Reset is Hardware or Software Reset ($\overline{RESET}$ = 0 or CR_{10} = 1).

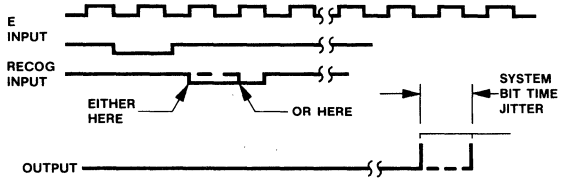
Counter recycling or re-initialization occurs when a negative transition of the clock input is recognized after the counter has reached an all-"0" state. In this case, data is transferred from the latches to the counter.

Asynchronous Input/Output Lines

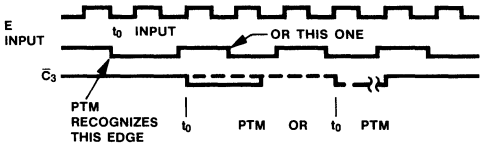
Each of the three timers within the PTM has external clock and gate inputs as well as a counter output line. The inputs are high impedance, TTL-compatible lines and outputs are capable of driving two standard TTL loads.

Clock Inputs ($\overline{C}_1$, $\overline{C}_2$ and $\overline{C}_3$)
Input pins $\overline{C}_1$, $\overline{C}_2$ and $\overline{C}_3$ will accept asynchronous TTL voltage level signals to decrement timers 1, 2 and 3, respectively. The HIGH and LOW levels of the external clocks must each be stable for at least one system clock period plus the sum of the set-up and hold times for the inputs. The asynchronous clock rate can vary from dc to the limit imposed by Enable (System $\phi 2$) set-up and hold time.

The external clock inputs are clocked in by Enable (System $\phi 2$) pulses. Three enable periods are used to synchronize and process the external clock. The fourth Enable pulse decrements the internal counter. This does not affect the input frequency, it merely creates a delay between a clock input transition and internal recognition of that transition by the PTM. All references to $\overline{C}$ inputs in this document relate to internal recognition of the input transition. Note that a clock HIGH or LOW level which does not meet set-up and hold time specifications may require an additional Enable pulse for recognition. When observing recurring events, a lack of synchronization will result in jitter being observed on the output of the PTM when using asynchronous clocks and gate input signals. There are two types of jitter. System jitter is the result of the input signals being out of synchronization with the Enable input (System $\phi 2$), permitting signals with marginal set-up and hold time to be recognized by either the bit time nearest the input transition or the subsequent bit time.



Input jitter can be as great as the time between input signal negative going transitions plus the system jitter, if the first transition is recognized during one system cycle, and not recognized the next cycle, or vice versa.



External clock input $\overline{C}_3$ represents a special case when timer 3 is programmed to utilize its optional ÷8 prescaler mode. The maximum input frequency and allowable duty cycles for this case are specified in the *AC Characteristics* table. The output of the ÷8 prescaler is treated in the same manner as the previously discussed clock inputs. That is, it is clocked into the counter by Enable pulses, is recognized on the fourth Enable pulse (provided set-up and hold time requirements are met), and must produce an output pulse at least as wide as the sum of an enable period, set-up and hold times.

Gate Inputs ($\overline{G}_1$, $\overline{G}_2$, $\overline{G}_3$)
Input lines $\overline{G}_1$, $\overline{G}_2$ and $\overline{G}_3$ accept asynchronous TTL-compatible signals which are used as triggers or clock gating functions to timers 1, 2 and 3, respectively. The gating inputs are clocked into the PTM by the Enable (System $\phi 2$) signal in the same manner as the previously discussed Clock inputs. That is, a Gate transition is recognized by the PTM on the fourth Enable pulse (provided set-up and hold time requirements are met), and the HIGH or LOW levels of the Gate input must be stable for at least one system clock period plus the sum of the set-up and hold times. All references to $\overline{G}$ transition in this document relate to internal recognition of the input transition.

The Gate inputs of all timers directly affect the internal 16-bit counter. The operation of $\overline{G}_3$ is therefore independent of the ÷8 prescaler selection.

Timer Outputs (O_1, O_2, O_3)
Timer outputs O_1, O_2 and O_3 are capable of driving up to two TTL loads and produce a defined output waveform for either continuous or single-shot timer modes. Output waveform definition is accomplished by selecting either single 16-bit or dual 8-bit operating modes. The single 16-bit mode will produce a square-wave output in the continuous timer mode and will produce a single pulse in the single-shot timer mode. The dual 8-bit mode will produce a variable duty cycle pulse in both the continuous and single shot timer modes. One bit of each control register (CRX_7) is used to enable the corresponding output. If this bit is cleared, the output will remain LOW (V_{OL}) regardless of the operating mode.

The continuous and single-shot timer modes are the only ones for which output response is defined. Signals appear at the outputs (unless CRX_7 = "0") during frequency and pulse width comparison modes, but the actual waveform is not predictable in typical applications.

Timer Operating Modes

The F6840 has been designed to operate effectively in a wide variety of applications. This is accomplished by using three bits of each control register (CRX_3, CRX_4 and CRX_5) to define different operating modes of the timers. These modes are outlined in *Table 4*.

Table 4 Operating Modes

Control Register			
CRX_3	CRX_4	CRX_5	Timer Operating Mode
0	*	0	Continuous
0	*	1	Single-Shot
1	0	*	Frequency Comparison
1	1	*	Pulse Width Comparison

*Defines additional timer functions

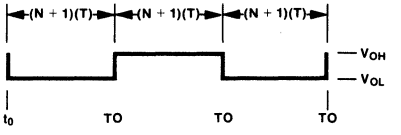
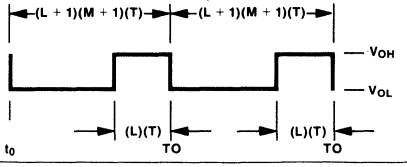
In addition to the four timer modes in *Table 4*, the remaining control register bit is used to modify counter initialization and enabling or interrupt conditions.

Continuous Operating Mode *(Table 5)*
Any of the timers in the PTM may be programmed to operate in a continuous mode by writing "0s" into bits 3 and 5 of the corresponding control register. Assuming that the timer output is enabled (CRX_7 = "1"), either a square wave or a variable duty cycle waveform will be generated at the timer output, OX. The type of output is selected via control register bit 2.

Either a timer reset ($CRX1_0$ = "1" or External $\overline{RESET}$ = "0") condition or internal recognition of a negative transition of the Gate input results in counter initialization. A write timer latches command can be selected as a counter initialization signal by clearing CRX_4.

In the dual 8-bit mode (CRX_2 = "1") [refer to the example in *Figure 1*] the MSB decrements once for every full countdown of the LSB + 1. When the

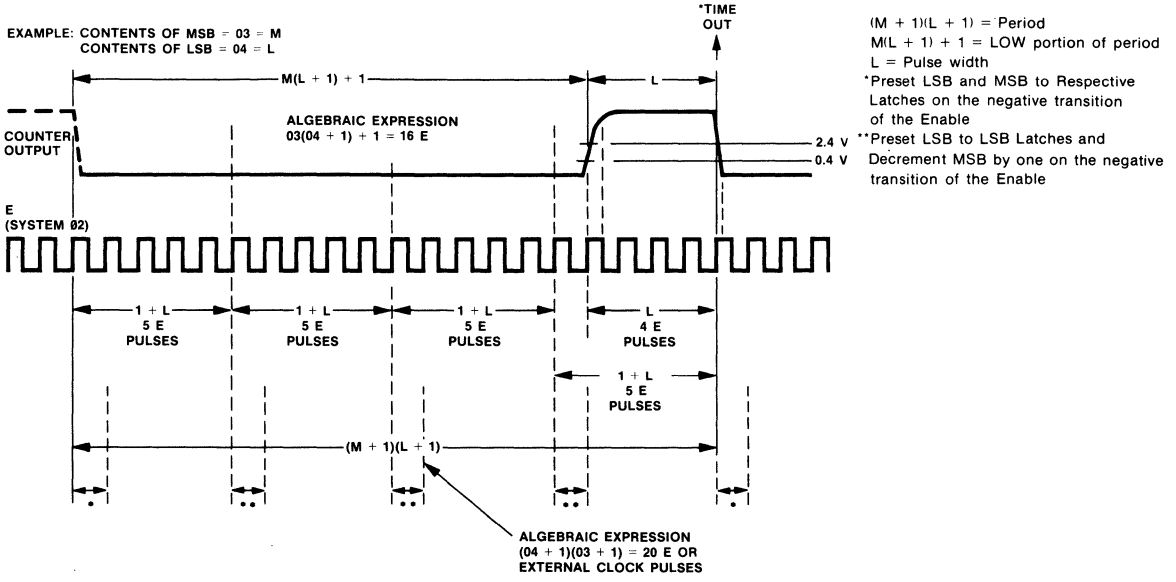
Table 5 Continuous Operating Modes, (CRX_3 = "0", CRX_5 = "0")

Control Register		Initialization/Output Waveforms	
CRX_2	CRX_4	Counter Initialization	*Timer Output (OX) (CRX_7 = "1")
0	0	$\overline{G}\downarrow$ + W + R	
0	1	$\overline{G}\downarrow$ + R	
1	0	$\overline{G}\downarrow$ + W + R	
1	1	$\overline{G}\downarrow$ + R	

$\overline{G}\downarrow$ = Negative transition of Gate input
W = Write Timer Latches Command
R = Timer Reset ($CR1_0$ = "1" or External $\overline{RESET}$ = "0")
N = 16-Bit Number in Counter Latch
L = 8-Bit Number in LSB Counter Latch
M = 8-Bit Number in MSB Counter Latch
T = Clock Input Negative Transitions to Counter
t_0 = Counter Initialization Cycle
TO = Counter Time-Out (All "0" Condition)

*All time intervals shown above assume the Gate ($\overline{G}$) and Clock ($\overline{C}$) signals are synchronized to Enable (System $\phi 2$) with the specified set-up and hold time requirements.

Fig. 1 Timer Output Waveforms Example



LSB = "0", the MSB is unchanged; on the next clock pulse the LSB is reset to the count in the LSB latches and the MSB is decremented by 1 (one). The output, if enabled, remains LOW during and after initialization and will remain LOW until the counter MSB is all "0s". The output will go HIGH at the beginning of the next clock pulse. The output remains HIGH until both the LSB and MSB of the counter are all "0s". At the beginning of the next clock pulse the defined time-out (TO) will occur and the output will go LOW. In the normal 16-bit mode the period of the output of the example in *Figure 1* would span 1546 clock pulses as opposed to the 20 clock pulses using the dual 8-bit mode.

The counter is enabled by an absence of a timer reset condition and a logic "0" at the Gate input. The counter will then decrement on the first clock signal recognized during or after the counter initialization cycle. It continues to decrement on each clock signal so long as $\overline{G}$ remains LOW and no reset condition exists. A counter time-out (the first clock after all counter bits = "0") results in the individual interrupt flag being set and re-initialization of the counter.

A special condition exists for the dual 8-bit mode (CRX_2 = "1") if L = "0". In this case, the counter will revert to a mode similar to the single 16-bit mode, except time-out occurs after M + 1 clock pulses. The output, if enabled, goes LOW during the counter initialization cycle and reverses state at each time-out. The counter remains cyclical (is re-initialized at each time-out) and the individual interrupt flag is set when time-out occurs. If M = L = "0", the internal counters do not change, but the output toggles at a rate of 1/2 the clock frequency.

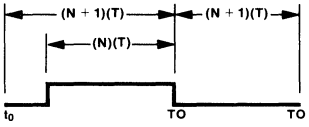
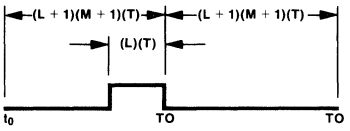
The discussion of the continous mode has assumed that the application requires an output signal. It should be noted that the timer operates in the same manner with the output disabled (CRX_7 = "0"). A read timer counter command is valid regardless of the state of CRX_7.

Single-Shot Timer Mode

This mode is identical to the continuous mode with three exceptions. The first of these is obvious from the name — the output returns to a LOW level after the initial time-out and remains LOW until another counter initialization cycle occurs. The waveforms available are shown in *Table 6*.

As indicated in *Table 6*, the internal counting mechanism remains cyclical in the single-shot mode. Each time-out of the counter results in the setting of an individual interrupt flag and re-initialization of the counter.

Table 6 Single-Shot Operating Modes, (CRX_3 = "0", CRX_7 = "1", CRX_5 = "1")

Control Register		Initialization/Output Waveforms	
CRX_2	CRX_4	Counter Initialization	Timer Output (OX)
0	0	$\overline{G}\downarrow + W + R$	
0	1	$\overline{G}\downarrow + R$	
1	0	$\overline{G}\downarrow + W + R$	
1	1	$\overline{G}\downarrow + R$	

Symbols are as defined in *Table 5*

The second major difference between the single-shot and continuous modes is that the internal counter enable is not dependent on the Gate input level remaining in the LOW state for the single-shot mode.

Another special condition is introduced in the single-shot mode. If L = M = "0" (Dual 8-bit) or N = "0" (Single 16-bit), the output goes LOW on the first clock received during or after counter initialization. The output remains LOW until the operating mode is changed or non-"0" data is written into the counter latches. Time-outs continue to occur at the end of each clock period.

The three differences between single-shot and continuous timer modes can be summarized as attributes of the single-shot mode:

1. Output is enabled for only one pulse until it is reinitialized.
2. Counter Enable is independent of Gate.
3. L = M = "0" or N = "0" disables output.

Aside from these differences, the two modes are identical.

Frequency Comparison or Period Measurement Mode (CRX_3 = "1", CRX_4 = "0")

The frequency comparison mode with CRX_5 = "1" is straightforward. If time-out occurs prior to the first negative transition of the Gate input after a counter initialization cycle, an individual interrupt flag is set. The counter is disabled, and a counter initialization cycle cannot begin until the interrupt flag is cleared and a negative transition on $\overline{G}$ is detected.

Time Interval Modes

The time interval modes are provided for those applications which require more flexibility of interrupt generation and counter initialization. Individual interrupt flags are set in these modes as a function of both counter time-out and transitions of the Gate input. Counter initialization is also affected by interrupt flag status.

The output signal is not defined in any of these modes, but the counter does operate in either single 16-bit or dual 8-bit modes as programmed by CRX_2. Other features of the time interval modes are outlined in *Table 7*.

If CRX_5 = "0", as shown in *Table 7* and *Table 8*, an interrupt is generated if the Gate input returns LOW prior to a time-out. If counter time-out occurs first, the counter is recycled and continues to decrement. A bit is set within the timer on the initial time-out which precludes further individual interrupt generation until a new counter initialization cycle has been completed. When this internal bit is set, a negative transition of the Gate input starts a new counter initialization cycle. (The condition of $\overline{G}\downarrow \cdot \overline{I} \cdot TO$ is satisfied, since a time-out has occurred and no individual interrupt has been generated.)

Any of the timers within the PTM may be programmed to compare the period of a pulse (giving the frequency after calculations) at the Gate input with the time period required for counter time-out. A negative transition of the Gate input enables the counter and starts a counter

Table 7 Timer Interval Modes, CRX_3 = "1"

CRX_4	CRX_5	Application	Condition for Setting Individual Interrupt Flag
0	0	Frequency Comparison	Interrupt Generated if Gate Input Period (1/F) is less than Counter Time-Out (TO)
0	1	Frequency Comparison	Interrupt Generated if Gate Input Period (1/F) is greater than Counter Time-Out (TO)
1	0	Pulse Width Comparison	Interrupt Generated if Gate Input "Down Time" is less than Counter Time-Out (TO)
1	1	Pulse Width Comparison	Interrupt Generated if Gate Input "Down Time" is greater than Counter Time-Out (TO)

Table 8 Frequency Comparison Mode, CRX_3 = "1", CRX_4 = "0"

Control Register Bit 5 (CRX_5)	Counter Initialization	Counter Enable Flip-Flop Set (CE)	Counter Enable Flip-Flop Reset (CE)	Interrupt Flag Set (1)
0	$\overline{G}\downarrow \cdot \overline{T} \cdot (\overline{CE} + TO \cdot CE) + R$	$\overline{G}\downarrow \cdot \overline{W} \cdot \overline{R} \cdot \overline{T}$	$W + R + I$	$\overline{G}\downarrow$ Before TO
1	$\overline{G}\downarrow \cdot \overline{T} + R$	$\overline{G}\downarrow \cdot \overline{W} \cdot \overline{R} \cdot \overline{T}$	$W + R + I$	TO Before $G\downarrow$

$\overline{T}$ represents the interrupt for a given timer.

Table 9 Pulse Width Comparison Mode, CRX_3 = "1", CRX_4 = "1"

Control Register Bit 5 (CRX_5)	Counter Initialization	Counter Enable Flip-Flop Set (CE)	Counter Enable Flip-Flop Reset (CE)	Interrupt Flag Set (I)
0	$\overline{G}\downarrow \cdot \overline{T} + R$	$\overline{G}\downarrow \cdot \overline{W} \cdot \overline{R} \cdot \overline{T}$	$W + R + I + G$	$\overline{G}\uparrow$ Before TO
1	$\overline{G}\downarrow \cdot \overline{T} + R$	$\overline{G}\downarrow \cdot \overline{W} \cdot \overline{R} \cdot \overline{T}$	$W + R + I + G$	TO Before $\overline{G}\uparrow$

G = Level sensitive recognition of Gate Input

initialization cycle — provided that other conditions as noted in *Table 8* are satisfied. The counter decrements on each clock signal recognized during or after counter initialization until an interrupt is generated, a write timer latches command is issued, or a Timer Reset condition occurs. It can be seen from *Table 8* that an interrupt condition will be generated if CRX_5 = "0" and the period of the pulse (single pulse or separately measured repetitive pulses) at the Gate input is less than the counter time-out period. If CRX_5 = "1", an interrupt is generated if the reverse is true.

Assume now with CRX_5 = "1" that a counter initialization has occurred and that the Gate input has returned LOW prior to counter time out. Since there is no individual interrupt flag generated, this automatically starts a new counter initialization cycle. The process will continue with frequency comparison being performed on each Gate input cycle until the mode is changed, or a cycle is determined to be above the predetermined limit.

Pulse Width Comparison Mode (CRX_3 = "1", CRX_4 = "1")

This mode is similar to the frequency comparison mode except that a positive, rather than negative, transition of the Gate input terminates the count. With CRX_5 = "0", an individual interrupt flag will be generated if the "0" level pulse applied to the Gate input is less than the time period required for counter time-out. With CRX_5 = "1", the interrupt is generated when the reverse condition is true.

As can be seen in *Table 9*, a positive transition of the Gate input disables the counter. With CRX_5 = "0", it is therefore possible to obtain directly the width of any pulse causing an interrupt. Similar data for other time interval modes and conditions can be obtained, if two sections of the PTM are dedicated to the purpose.

Absolute Maximum Ratings

Supply Voltage	-0.3 V, +7.0 V
Input Voltage	-0.3 V, +7.0 V
Operating Temperature Range — T_L to T_H	
F6840P,S/F68A40P,S/F68B40P,S	0° C, +70° C
F6840CP,CS/F68A40CP,CS	-40° C, +85° C
F6840DL	-55° C, +85° C
F6840DM	-55° C, +125° C
Storage Temperature Range	-55° C, +150° C
Thermal Resistance	
Plastic Package	115° C/W
Ceramic Package	60° C/W

Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted.

Symbol	Characteristic	Signal	Min	Typ	Max	Unit	Test Condition
V_{IH}	Input HIGH Voltage		2.0			V	
V_{IL}	Input LOW Voltage		-0.4		0.8	V	
I_{IN}	Input Leakage Current			1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	3-State (OFF State) Input Current	D_0-D_7		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
V_{OH}	Output HIGH Voltage	D_0-D_7 Other Outputs	2.4 2.4			V	I_{Load} = -205 μA, I_{Load} = 200 μA
V_{OL}	Output LOW Voltage	D_0-D_7 O_1-O_3, $\overline{IRQ}$			0.4 0.4	V	I_{Load} = 1.6 mA, I_{Load} = 3.2 mA
I_{LOH}	Output Leakage Current (OFF State)	($\overline{IRQ}$)		1.0	10	μA	V_{OH} = 2.4 V
P_D	Power Dissipation			470	700	mW	
C_{IN}	Input Capacitance	 D_0-D_7 All Other Inputs			 12.5 7.5	 pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance	O_1, O_2, O_3 $\overline{IRQ}$			10 5.0	 pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz

Bus Timing Characteristics
Read *(Figure 2)*

Symbol	Characteristic	F6840		F68A40		F68B40		Unit
		Min	Max	Min	Max	Min	Max	
$t_{cyc}E$	Enable Cycle Time	1.0	10	0.666	10	0.5	10	μs
PW_{EH}	Enable Pulse Width, HIGH	0.45	4.5	0.280	4.5	0.22	4.5	μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.280		0.21		μs
t_{AS}	Set-up Time, Address and R/$\overline{W}$ valid to enable positive transition	160		140		70		ns
t_{DDR}	Data Delay Time		320		220		180	ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable input		25		25		25	ns

Write *(Figure 3)*

$t_{cyc}E$	Enable Cycle Time	1.0	10	0.666	10	0.50	10	μs
PW_{EH}	Enable Pulse Width, HIGH	0.45	4.5	0.280	4.5	0.22	4.5	μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.280		0.21		μs
t_{AS}	Set-up Time, Address and R/$\overline{W}$ valid to enable positive transition	160		140		70		ns
t_{DSW}	Data Set-up Time	195		80		60		ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable input		25		25		25	ns

Fig. 2 Bus Read Timing Characteristics
(Read Information from PTM)

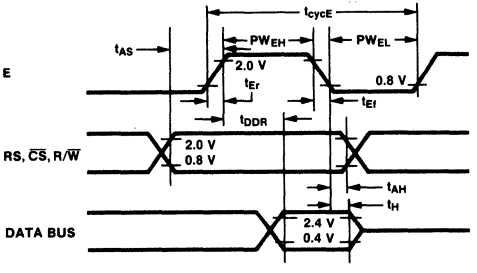
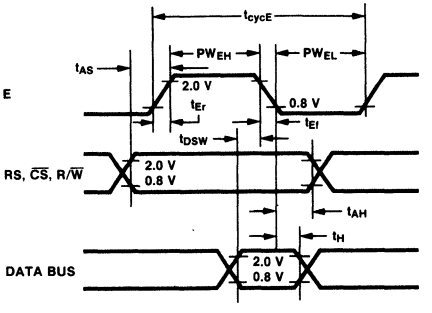


Fig. 3 Bus Write Timing Characteristics
(Write Information into PTM)



AC Characteristics *(Figures 4-8)*

Symbol	Characteristic		F6840 Min	F6840 Max	F68A40 Min	F68A40 Max	F68B40 Min	F68B40 Max	Unit
t_r, t_f	Input Rise and Fall Times	$\overline{C}$,$\overline{G}$ and $\overline{RESET}$		1.0*		0.666*		0.500*	μs
PW_L	Input Pulse Width LOW	$\overline{C}$,$\overline{G}$ and $\overline{RESET}$	$t_{cyc}E$ $+t_{su}$ $+t_{hd}$		$t_{cyc}E$ $+t_{su}$ $+t_{hd}$		$t_{cyc}E$ $+t_{su}$ $+t_{hd}$		ns
PW_H	Input Pulse Width HIGH	$\overline{C}$,$\overline{G}$	$t_{cyc}E$ $+t_{su}$ $+t_{hd}$		$t_{cyc}E$ $+t_{su}$ $+t_{hd}$		$t_{cyc}E$ $+t_{su}$ $+t_{hd}$		ns
t_{su}	Input Set-up Time (Synchronous Mode)	$\overline{C}$,$\overline{G}$ and $\overline{RESET}$ $\overline{C3}$ (÷8 Prescaler Mode only)	200		120		75		ns
t_{hd}	Input Hold Time (Synchronous Mode)	$\overline{C}$,$\overline{G}$ and $\overline{RESET}$ $\overline{C3}$ (÷8 Prescaler Mode only)	50		50		50		ns
	Output Delay, 01-03								
t_{co}	(V_{OH} = 2.4 V, Load B)	TTL		700		460		340	ns
t_{cm}	(V_{OH} = 2.4 V, Load D)	MOS		450		450		340	ns
t_{cmos}	(V_{OH} = 0.7 V_{DD}, Load D)	CMOS		2.0		1.35		1.0	μs
t_{IR}	Interrupt Release Time			1.2		0.9		0.7	μs

*t_r and $t_f \leq 1 \times$ Pulse Width or 1.0 μs, whichever is smaller.

Fig. 4 Input Pulse Width Low

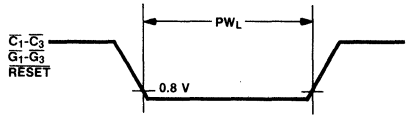


Fig. 5 Input Pulse Width High

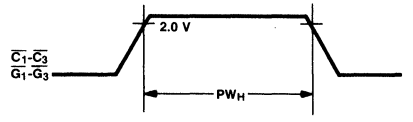


Fig. 6 Input Set-up and Hold Times

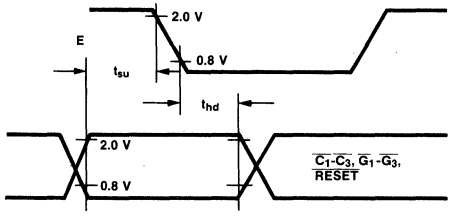


Fig. 7 Output Delay

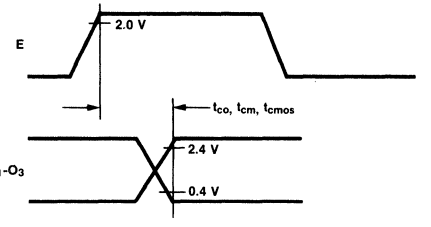


Fig. 8 $\overline{IRQ}$ Release Time

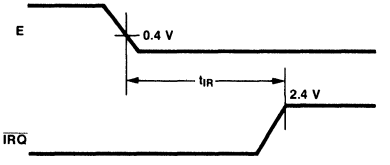
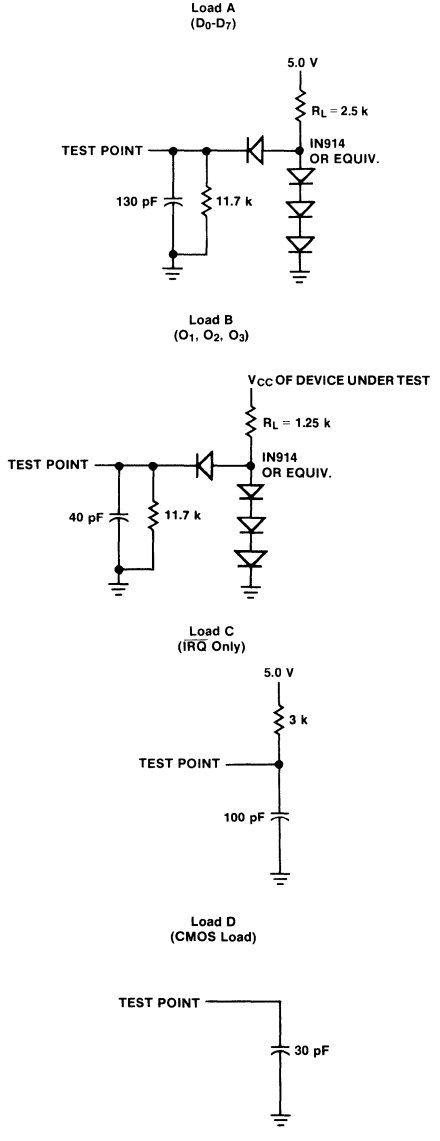


Fig. 9 Bus Timing Test Loads



Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6840P,S	0° C to +70° C
	F6840CP,CS	-40° C to +85° C
	F6840DL	-55° C to +85° C
	F6840DM	-55° C to +125° C
1.5 MHz	F68A40P,S	0° C to +70° C
2.0 MHz	F68B40P,S	0° C to +70° C

P = Plastic package, S = Ceramic package

5

FAIRCHILD
A Schlumberger Company

F6844
Direct Memory Access Controller

Microprocessor Product

Description

The F6844 Direct Memory Access Controller (DMAC) transfers data directly between memory and peripheral device controllers. In bus-organized systems, such as those based on the F6800 microprocessor, the DMAC, rather than the MPU, controls the address and data buses.

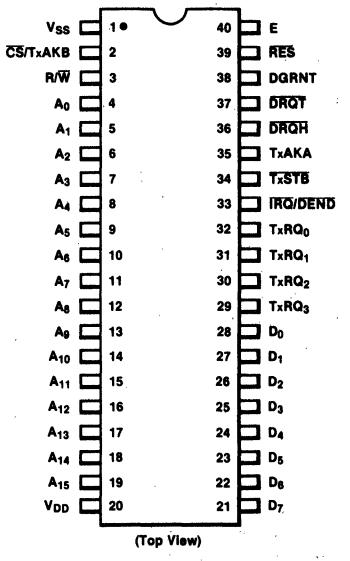
The DMAC bus interface includes select, read/write, interrupt, transfer request/grant, and bus interface logic to permit data transfer over an 8-bit bidirectional data bus. The F6844 functional configuration is programmed through the data bus. The internal structure provides for control and handling of four individual channels, each of which is separately configured. Programmable control registers provide control for the transfer location and length, individual channel control and transfer mode configuration, priority of servicing, data chaining, and interrupt control. Status and control lines serve the peripheral controllers.

The mode of transfer for each channel can be programmed as cycle-stealing or burst transfer.

Typical applications include use with the F6856 Synchronous Protocol Communications Controller, the F6854 Advanced Data Link Controller, and the F68488 IEEE-488 Bus Controller.

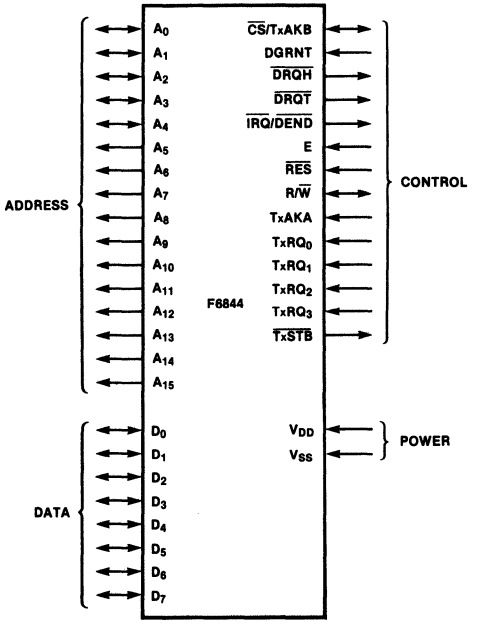
- **Four DMA Channels, Each Having a 16-Bit Address Register and a 16-Bit Byte Count Register**
- **2M Byte/Sec Maximum Data Transfer Rate**
- **Selection of Fixed or Rotating Priority Service Control**
- **Separate Control Bits for Each Channel**
- **Data Chain Function**
- **Address Increment or Decrement Update**
- **Programmable Interrupts and DMA End to Peripheral Controllers**

Connection Diagram
40-Pin DIP



(Top View)

F6844 Signal Functions



Functional Description

The DMAC has 15 addressable registers, of which eight are 16 bits in length (see *Figure 1*). Each channel has a separate address register and a byte count register, each of which is 16 bits. There are four channel control registers with three common general control registers (priority, interrupt, and data chain).

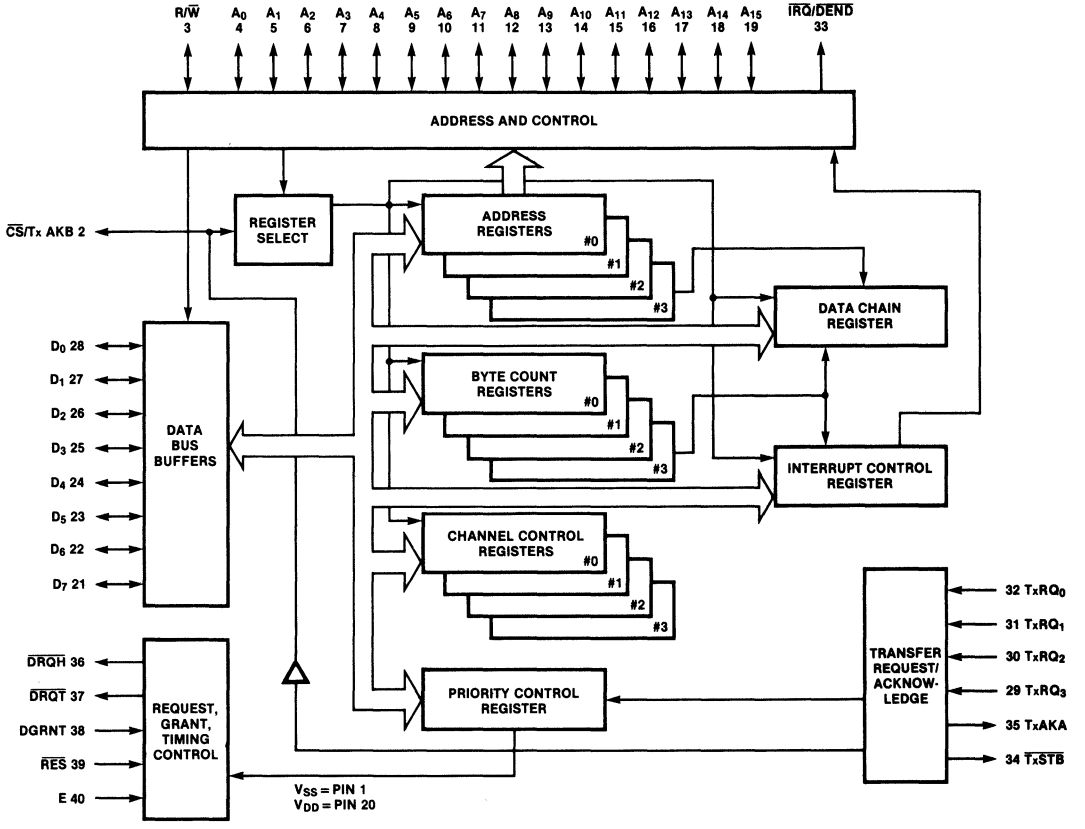
To prepare a channel for direct memory access (DMA), the address registers must be loaded with the starting memory address and the byte count register loaded with the number of bytes to be transferred. The bits in the channel control register establish the direction of the transfer, the mode, and the address increment or decrement after each cycle. Each channel can be set for one of three transfer modes: three-state control (TSC) steal, halt steal, or halt burst. Two read-only status bits in the channel control register indicate when the channel is busy transferring data and when the DMA transfer is complete.

The priority control register enables the transfer requests from the peripheral controllers and establishes either a fixed priority or rotating priority scheme of servicing these requests. When the DMA transfer for a channel is complete (the byte count register is zero), a DMA end ($\overline{\text{DEND}}$) signal is directed to the peripheral controller and an interrupt request ($\overline{\text{IRQ}}$) goes to the MPU. The interrupt control register enables these interrupts; the $\overline{\text{IRQ}}$/$\overline{\text{DEND}}$ flag bit is read from this register.

Chaining of data transfers is controlled by the data chain register. When enabled, the contents of the address and byte count registers for channel 3 are put into the registers of the channel selected for chaining as its byte count register becomes zero. This allows for repetitively reading or writing a block of memory.

During the DMA mode, the DMAC controls the address bus and data bus for the system as well as provides the R/$\overline{\text{W}}$ line and a signal to be used as valid memory address (VMA). When a peripheral device controller desires a DMA transfer, it issues a transfer request. Assuming this request is enabled and meets the test of highest priority, the DMAC issues a DMA request. When the DMAC receives the DMA grant (DGRNT) input, it gives a transfer acknowledge (TxAKA or TxAKB) to the peripheral device controller, at which time the data is transferred. When the channel byte count register equals zero, the transfer is complete, a $\overline{\text{DEND}}$ is given to the peripheral device controller, and an $\overline{\text{IRQ}}$ is given to the MPU.

Fig. 1 Block Diagram



5

Signal Descriptions

The F6844 input and output signals are described in *Table 1.*

Table 1 F6844 Signal Functions

Mnemonic	Pin No.	Name	Description
Address			
A_0-A_4	4-8	Address	In the MPU mode, the signals are high-impedance inputs used to address the DMAC registers. In the DMA mode, these outputs are set to the contents of the address register for the channel being processed.
A_5-A_{15}	9-19	Address	These output lines are in the high-impedance state during the MPU mode. In the DMA mode, these lines are outputs that are set to the contents of the address register for the channel being processed.
Data			
D_0-D_7	28-21	Bidirectional Data	The eight bidirectional lines provide data transfer between the DMAC and the MPU. The data bus output drivers are three-state devices that remain in the high-impedance state except when the MPU performs DMAC read operations.
Control			
$\overline{CS}$/TxAKB	2	Chip Select/ Transfer Acknowledge B	This signal is an output in the four-channel mode during the DMA transfer. At all other times, it is a high-impedance, TTL-compatible input used to address the DMAC. The DMAC is selected when $\overline{CS}$/TxAKB is low. Valid memory address (VMA) must be used in generating this input to prevent false selects. Transfers of data to and from the DMAC are then controlled by the E, read/write, and A_0-A_4 address lines. In the four-channel mode, when TxAKB is needed, the $\overline{CS}$ gate must have an open-collector output (a pull-up resistor should not be used). In the two-channel mode, $\overline{CS}$/TxAKB is always an input.
DGRNT	38	DMA Grant	A high-impedance input signal to the DMAC, providing control of the system buses. In the three-state control (TSC) steal mode, the signal comes from the system clock drive circuit (DMA grant), indicating that the clock is being stretched. For the halt steal or halt burst mode, this signal is the bus available (BA) from the MPU, indicating that the MPU has halted and transferred control of its buses to the DMAC. For a design involving TSC steal and halt mode transfers, this input must be the logical OR of the clock-driven DMA grant and the MPU BA.
$\overline{DRQH}$	36	DMA Request Halt Steal	This active-low output requests a DMA transfer for a channel programmed for the halt steal or halt burst transfer mode. The signal is connected directly to the MPU HALT input and remains low until the last byte transfer has begun.

Table 1 F6844 Signal Functions (Cont.)

Mnemonic	Pin No.	Name	Description
$\overline{DRQT}$	37	DMA Request Three-State Control Steal	This active-low output requests a DMA transfer for a channel configured for the TSC steal transfer mode. The signal is connected to the system clock driver, requesting a $\phi 1$ clock stretch. It remains in the low state until the transfer has begun.
E	40	Direct Memory Access	The DMAC register I/O transfers, channel request line sampling, and gating of other control signals to the system are done internally in conjunction with the E high-impedance input. This input must be the system memory clock (a nonstretched E clock).
$\overline{IRQ}/\overline{DEND}$	33	Interrupt Request/ DMA End	A TTL-compatible, active-low output used to interrupt the MPU and to signal the peripheral controller that the data block transfer has ended. If the interrupt has been enabled, the $\overline{IRQ}/\overline{DEND}$ line goes low after the last DMA cycle of a transfer. An open-collector gate must be connected to DGRNT and $\overline{IRQ}/\overline{DEND}$ to prevent false interrupts from the $\overline{DEND}$ signal when interrupts are not enabled.
$\overline{RES}$	39	Reset	The $\overline{RES}$ input resets the DMAC from an external source. In the low state, the $\overline{RES}$ input causes all registers, except address and byte count, to be reset to the logic 0 state. This disables all transfer requests, masks all interrupts, disables the data chain function, and puts each channel control register into the condition of memory write, halt steal transfer mode, and address increment.
R/$\overline{W}$	3	Read/Write	A TTL-compatible signal that is a high-impedance input in the MPU mode and an output in the DMA mode. In the MPU mode, it controls the direction of data flow through the DMAC input/output data bus interface. When read/write is high (MPU read cycle) and the chip is selected, DMAC data output buffers are turned on and a selected register is read. When it is low, the DMAC output drivers are turned off and the MPU writes into a selected register. In the DMA mode, read/write is an output to drive the memory and peripheral controllers. Its state is determined by bit 0 of the channel control register for the channel being serviced. When read/write is high, the memory is written into the peripheral controller. When it is low, the peripheral controller is read and its data stored in the memory. In the DMA mode, the DMAC data buffers are off, so data is not available on the data bus (D_0-D_7).

Table 1 F6844 Signal Functions (Cont.)

Mnemonic	Pin No.	Name	Description
TxAKA	35	Transfer Acknowledge A	This signal is a TTL-compatible output used in conjunction with the $\overline{CS}$/TxAKB line to select the channel to be strobed for transfer, and to give the DMA end signal. In the two-channel mode, only TxAKA is used to select channel 0 or 1, and $\overline{CS}$/TxAKB is always an input.
$TxRQ_0$-$TxRQ_3$	32-29	Transfer Request	Each of the four channels has its own high-impedance input request for transfer line. The peripheral controller requests a transfer by setting its TxRQ line high (a logic 1). The lines are sampled according to the priority and enabling established in the priority control register. In the halt steal mode, and the first byte of the halt burst mode, the TxRQ signals are tested on the positive edge of E and the highest priority channel is strobed. Once strobed, the TxRQs are not tested again until that channel's data transfer is finished. In the succeeding bytes of the halt burst mode transfer, the TxRQ is tested on the negative edge of E, and data is transferred on the next E cycle if the TxRQ signal is high.
$\overline{TxSTB}$	34	Transfer Strobe	This output signal is an acknowledgement to the peripheral controller, and controls transfer of data to or from memory. The transfer strobe is also used as the VMA signal in the DMA mode. In a one-channel system, $\overline{TxSTB}$ can be inverted and run to the peripheral controller acknowledge input. In a two- or four-channel system, $\overline{TxSTB}$ enables the decode of TxAKA and $\overline{CS}$/TxAKB to select the device controller to be acknowledged.
Power			
V_{DD}	20	Power Supply	Nominal +5 Vdc
V_{SS}	1	Ground	Common power and signal return

DMAC Register Descriptions

The 15 registers in the DMAC are read/write registers, although some of the bits are read-only status bits.

Address Registers
Each channel has its own individual 16-bit address register. Before a DMA transfer is begun, the starting address for the transfer must be loaded into the address register. Depending on the state of bit 3 of the channel control register, the address register is decremented or incremented after each byte transfer.

Byte Count Registers
Each channel also has its own byte count register. Before the DMA transfer, this register must be loaded with the number of bytes to be transferred. Since it is 16 bits in length, the transfer can be up to 65,536 bytes of data. The byte count register is decremented at the beginning of each DMA cycle.

Channel Control Registers
The control of each channel's DMA transfer is programmed into its channel control register. Bits 4 and 5 are unused.

Bit 0, Read/Write (R/$\overline{W}$)—The direction of the DMA transfer is controlled by this bit. When it is high, the peripheral controller reads the memory. When it is low, the transfer is in the opposite direction, thus writing into the memory. The system R/$\overline{W}$ line is in the same state as this R/$\overline{W}$ bit in the DMA mode. The device controller must change the sense of its R/$\overline{W}$ input during the DMA mode.

Bit 1, Burst/Steal—This bit, along with bit 2, selects the mode of the DMA transfer. With bit 1 high, the burst mode is selected. A low selects the steal mode.

B2, TSC/Halt—This bit helps select the mode of DMA transfer. When the bit is high, the TSC mode is selected. When low, the halt mode is selected. A TSC burst mode is illegal for F6800-family processors due to restrictions on $\phi 1$ clock stretching for these products.

The mode selection for bits 1 and 2 is as follows:

Bit 2	Bit 1	DMA Transfer Mode
0	0	Halt Steal
0	1	Halt Burst
1	0	TSC Steal
1	1	(Illegal)

Bit 3, Address Up/Down—Bit 3 controls the change in the address register for each DMA cycle. If this bit is low, the address register is incremented each time the byte count register decrements. If the bit is high, the address register is decremented.

Bit 6, Busy/Ready Flag—The busy/ready flag is a read-only status bit that indicates a DMA transfer is in process on that channel. This bit goes high at the beginning of the transfer and remains high until the $\overline{IRQ}/\overline{DEND}$ has been low for one cycle (DMA end). The bit is then reset and the channel can again be configured for transfer.

Bit 7 DMA End (DEND) Flag—The DEND bit indicates that a DMA block transfer has ended. This bit is set at the same time the busy/ready flag is reset. The DEND bit is reset by the MPU reading the channel control register. This bit causes an interrupt if enabled in the interrupt control register.

5

Priority Control Register
The priority control register establishes priority and enables the transfer requests. Bits 4, 5, and 6 are unused.

Bits 0-3, Request Enable (RE_{0-3})—The four channels are individually enabled by setting the respective RE bit high. A low on any of these bits disables recognition of the transfer request for that channel. The bit number represents the channel number; e.g., bit 2 is channel 2.

Bit 7, Rotate Control—The DMAC priority service routine is selected by this rotate control bit. When it is low, the fixed mode is selected. Channel 0 has the highest priority, channel 1 the next highest, etc. When this bit is high, a rotating routine is used: initially, it is the same as in the fixed mode, but once a channel has been serviced, it moves to the lowest priority and those that were below it advance to the next higher priority.

Interrupt Control Register
An interrupt is caused by a channel completing its DMA block transfer. The DEND (channel control register bit 7) flags this condition for each channel. Bits 4, 5, and 6 are unused.

Bits 0-3, $\overline{IRQ}/\overline{DEND}$ Enable (DIE_{0-3})—Each channel is separately enabled to cause the interrupt. A high enables an interrupt from the channel; a low masks the interrupt. The bit number corresponds to the channel number; e.g., bit 2 is channel 2.

Bit 7, $\overline{\text{IRQ}}$/$\overline{\text{DEND}}$ Flag—This read-only bit indicates an IRQ is requested of the MPU when the signal is high. If the interrupt is enabled (DIE is a 1) when a channel's DEND flag (channel control register bit 7) goes high, the $\overline{\text{IRQ}}$/$\overline{\text{DEND}}$ flag bit also goes high. It is reset by the MPU reading the channel control register that caused the interrupt.

Data Chain Register
Repetitive reading or writing of a block of memory can be done in the data chain function. A DMA transfer cannot be active on channel 3 during the data chain. Bits 4 through 7 are unused.

Bit 0, Data Chain Enable (DCE)—The data chain function is enabled when this bit is high.

Bits 1 and 2, Data Chain Channel Select A, B (DCA, DCB)—The channel updated by data chaining is selected by bits 1 and 2 as follows:

DCB Bit 2	DCA Bit 1	Channel #
0	0	0
0	1	1
1	0	2
1	1	(Illegal)

The data chain function is performed by transferring the contents of channel 3 address and byte count registers into the respective registers of the channel selected by bits 1 and 2. The transfer occurs during the cycle of E following the byte count register having decremented to zero.

Bit 3, Two/Four Channel Select (2/4)—Bit 3 configures the DMAC to handle two or four channels. When high, this bit selects the four-channel mode, in which the $\overline{\text{CS}}$/TxAKB becomes a chip select in the MPU mode and a transfer acknowledge B in the DMA mode. With bit 3 low, the two-channel mode is selected, and the $\overline{\text{CS}}$/TxAKB line is always a chip select, both for the MPU and the DMA mode.

Initialization
During a power-on sequence, the DMAC is reset through the $\overline{\text{RES}}$ input. All registers, except the address and byte count, are set to a logic 0 state. This disables all requests and the data chain function, while masking all interrupts. The address, byte count, and channel control registers must be programmed before the respective transfer request bit is enabled in the priority control register.

Transfer Modes
Three methods are used for a DMA transfer, determined by the data transfer rate required, the number of channels attached, and the hardware complexity allowable. Refer to *Figure 2 (TSC Steal Mode), Figure 3 (Halt Steal Mode)*, and *Figure 4 (Halt Burst Mode)* for an illustration of the three DMA transfer methods.

Two of the modes, TSC steal and halt steal, are accomplished by cycle stealing from the MPU. Cycle stealing, in the TSC steal mode, is initiated by the DMAC bringing the $\overline{\text{DRQT}}$ line low. This line goes to the system clock driver, which returns a high on DGRNT on the rising edge of the system $\phi 1$ clock. The DGRNT signal must cause the address control and data lines to go to the high-impedance state, at which time the DMAC supplies the address from the address register of the requesting channel. It also supplies the R/$\overline{\text{W}}$ signal as determined from the channel control register. After one byte is transferred, control is restored to the MPU. This method stretches the $\phi 1$ and $\phi 2$ clocks while the DMAC uses the memory (see *Figure 5*).

Fig. 2 TSC Steal Mode Timing

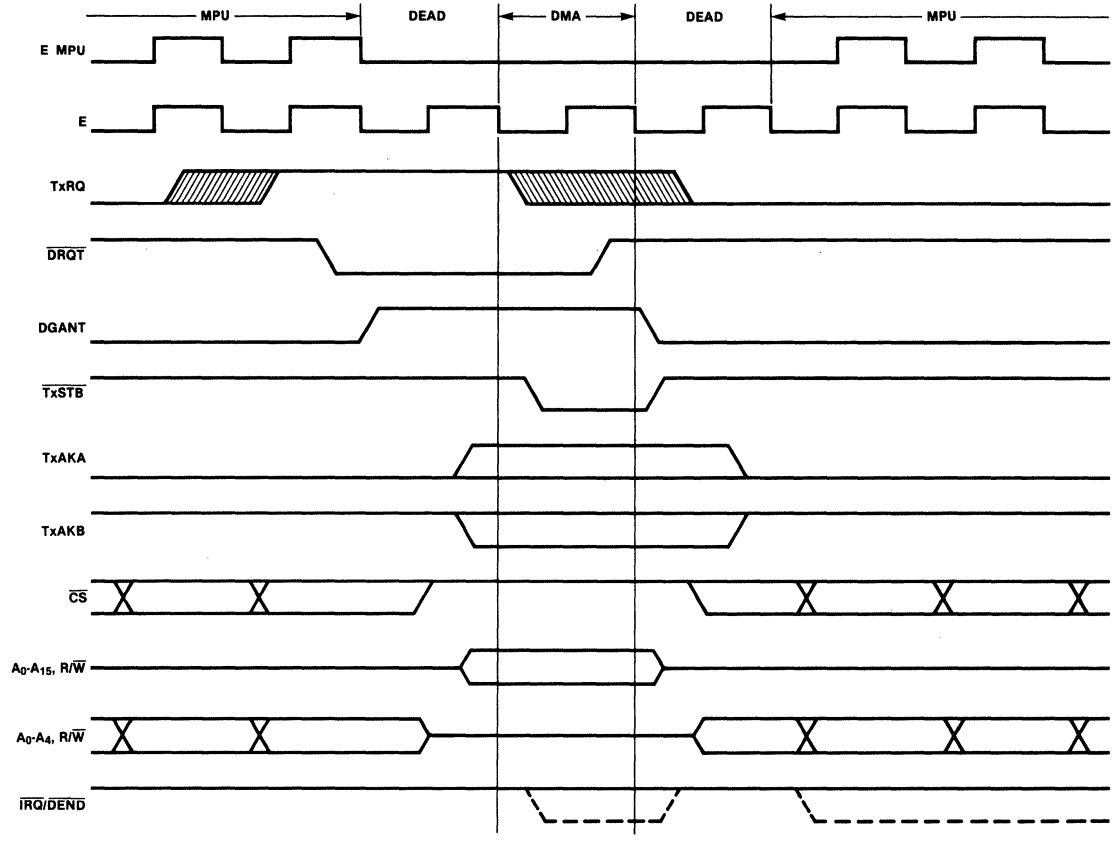


Fig. 3 Halt Steal Mode Timing

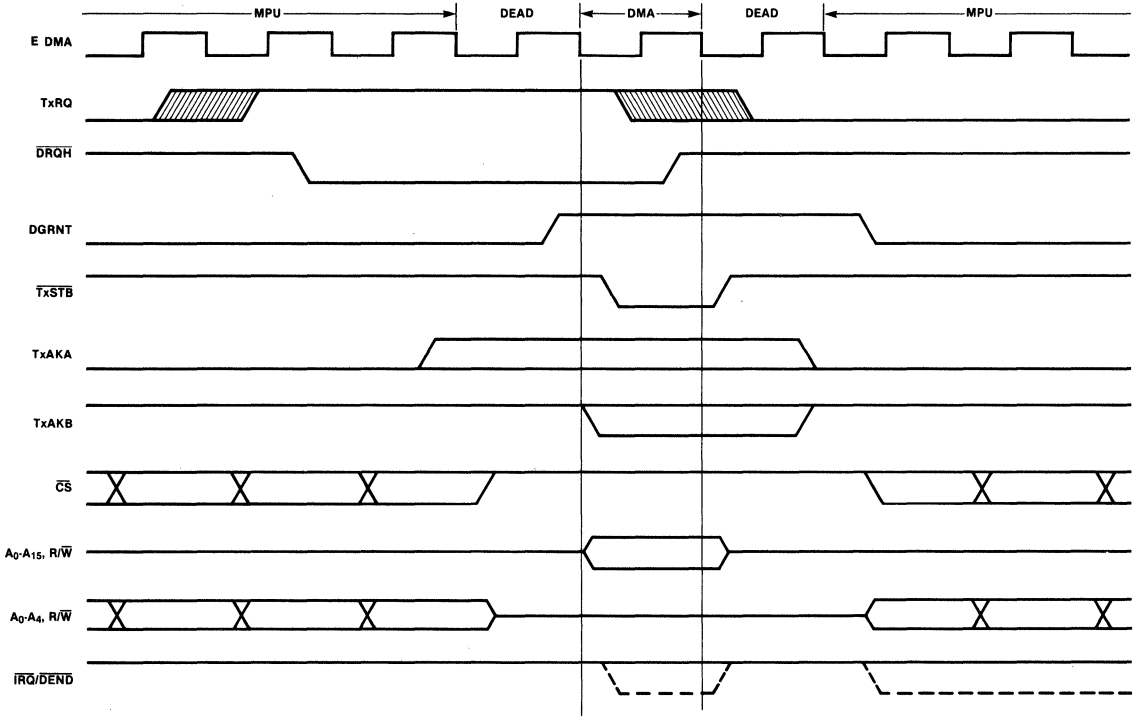
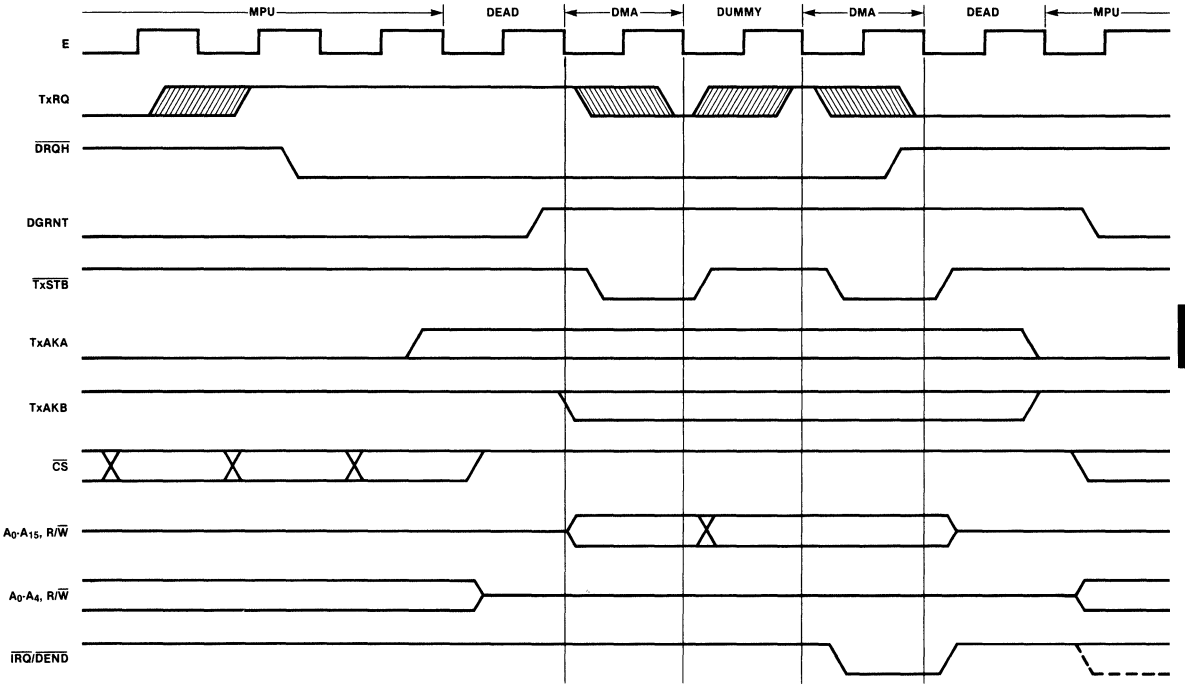


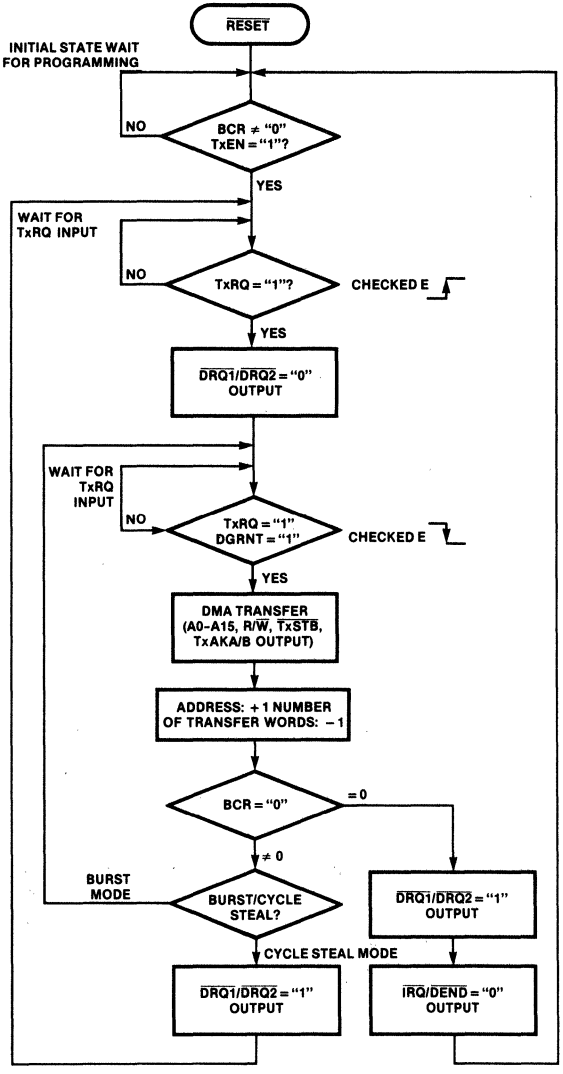
Fig. 4 Halt Burst Mode Timing



The second mode employing cycle stealing is the halt steal mode. This method actually halts the MPU instead of stretching the $\phi 1$ clock for the transfer period. This mode is initiated by the DMAC bringing the $\overline{DRQH}$ line low. This line connects to the MPU $\overline{HALT}$ input. The MPU bus available (BA) line is the DGRNT input to the DMAC. While the MPU is halted, its address bus, data bus, and R/$\overline{W}$ lines are in the high-impedance state. The DMAC supplies the address and R/$\overline{W}$ line. After one byte is transferred, the $\overline{HALT}$ line is returned high and the MPU regains control. In this mode, the MPU stops internal activity and is removed from the system while the DMAC uses the memory.

The third mode of transfer is the halt burst. This mode is similar to the halt steal mode, except that the transfer does not stop with one byte. The MPU is halted while an entire block of data is transferred. When the channel byte count register equals zero, the transfer is complete and control is returned to the MPU. This mode gives the highest data transfer rate, at the expense of the MPU being inactive during the transfer period.

Fig. 5 Flowchart of DMAC Operation



DMAC Programming Model

The following programming model outlines channel preparation for DMA transfer, request enabling, data chain register programming, and register descriptions (see *Table 2*).

Table 2 DMAC Programming Model

Register	Address (Hex)	Register Content							
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Channel Control	1x*	DMA End ($\overline{DEND}$) Flag	Busy/Ready Flag	Not Used	Not Used	Address Up/Down	TSC/ Halt	Burst/ Steal	Read/Write (R/$\overline{W}$)
Priority Control	14	Rotate Control	Not Used	Not Used	Not Used	Request Enable #3 (RE_3)	Request Enable #2 (RE_2)	Request Enable #1 (RE_1)	Request Enable #0 (RE_0)
Interrupt Control	15	$\overline{IRQ}/\overline{DEND}$ Flag	Not Used	Not Used	Not Used	$\overline{IRQ}/\overline{DEND}$ Enable #3 (DIE_3)	$\overline{IRQ}/\overline{DEND}$ Enable #2 (DIE_2)	$\overline{IRQ}/\overline{DEND}$ Enable #1 (DIE_1)	$\overline{IRQ}/\overline{DEND}$ Enable #0 (DIE_0)
Data Chain	16	Not Used	Not Used	Not Used	Not Used	Two/Four Channel Select (2/4)	Data Chain Channel Select B	Data Chain Channel Select A	Data Chain Enable

*The x represents the binary equivalent of the channel desired.

Channel Control Register

$\overline{DEND}$ — Bit 7— Is set at end of DMA block transfer; reset by MPU reading the channel control register.

Busy/Ready Flag — Bit 6— Status bit set when in transfer; cleared after DMA end.

Address Up/Down — Bit 3— High = decrement address register for each byte; low = increment.

TSC/Halt — Bit 2— High = select TSC mode; low = halt modes.

Burst/Steal — Bit 1— High = select burst mode; low = steal modes.

R/$\overline{W}$ — Bit 0— High = device controller reads memory; low = write into memory.

Priority Control Register

Rotate Control — Bit 7— High = use rotate routine; low = fixed: 0, 1, 2, 3 priority.

RE_{0-3} — Bits 0-3—High = enable transfer request for the channel; low = request disabled.

Interrupt Control Register

$\overline{IRQ}/\overline{DEND}$ Flag — Bit 7— This flag is set by $\overline{DEND}$ in channel control registers when enabled; reset by reading the register that caused it to be set.

DIE_{0-3} — Bits 0-3—High = enable $\overline{IRQ}$ by $\overline{DEND}$ for the channel; low = $\overline{IRQ}$ masked.

Data Chain Register

Two/Four Channel — Bit 3— High = 4-channel mode; low = 2-channel.

Data Chain Channel Select — Bits 2, 1—Binary equivalent of channel to be updated by chaining.

Data Chain Enable — Bit 0— High = enable data chain function; low = disabled.

Preparation of a channel for a DMA transfer requires:

1. Load the starting address into the address register.
2. Load the number of bytes into the byte count register.
3. Program the channel control register for the transfer characteristics: direction (bit 0), mode (bits 1 and 2), and the address update (bit 3).

The channel is now configured. To enable the transfer request, set the appropriate enable bit (bits 0-3) of the priority control register, as well as the rotate control bit.

If an interrupt on $\overline{\text{DEND}}$ is desired, the enable bit (bits 0-3) of the interrupt control register must be set.

If data chaining for the channel is necessary, it is programmed into the data chain register and the appropriate data must be written into the address and byte count registers for channel 3.

A comparison of the response times and maximum transfer rates is shown below. The values shown are for a system clock rate of 1 MHz.

Mode	Response Time (μs)	Maximum Transfer Rate (μs/byte)
Halt Burst	3.5-15.5*	1
Halt Steal	3.5-15.5*	5-15*
TSC Steal	2.5-3.5	4

*These values depend upon the cycle in process.

The two 8-bit bytes that form the registers in *Table 3* are placed in consecutive memory locations, making it very easy to use the MPU index register in programming them.

Table 3 Address and Byte Count Registers

Register	Channel	Address (Hex)
Address High	0	0
Address Low	0	1
Byte Count High	0	2
Byte Count Low	0	3
Address High	1	4
Address Low	1	5
Byte Count High	1	6
Byte Count Low	1	7
Address High	2	8
Address Low	2	9
Byte Count High	2	A
Byte Count Low	2	B
Address High	3	C
Address Low	3	D
Byte Count High	3	E
Byte Count Low	3	F

System Description

The DMAC hardware configuration is designed for a one-, two-, or four-channel system.

$\overline{IRQ}$, $\overline{DEND}$, TxAK Generation

Derivation of the $\overline{IRQ}$, $\overline{DEND}$, and TxAK signals for one-, two-, and four-channel DMA is shown in *Figures 6, 7,* and *8*. The $\overline{IRQ}$ signal, if enabled, is asserted by the DMA to interrupt the MPU whenever a DMA block transfer is completed. The TxAK signal is asserted during each DMA cycle and is used to handshake with a peripheral controller each time a DMA byte transfer occurs. The $\overline{DEND}$ signal is used to handshake with a peripheral controller each time a DMA block transfer is complete.

Each circuit uses DMA GRANT to demultiplex the $\overline{IRQ}/\overline{DEND}$ DMAC output to ensure that the system $\overline{IRQ}$ is asserted at the proper time, only during MPU operation. Whenever DMA GRANT is high, $\overline{IRQ}$ is negated.

The circuits also generate DEND and TxAK for the proper channel, gated by TxSTB.

The one-channel DMA mode requires no channel decoding, so for this mode TxAK is derived from $\overline{TxSTB}$ directly, and $\overline{TxSTB}$ is used to demultiplex the $\overline{IRQ}/\overline{DEND}$ output for $\overline{DEND}$ generation.

The two-channel mode circuit is similar to the one-channel circuit but uses TxAKA to identify the active channel and generate the appropriate channel signal.

Fig. 6 One-channel Operation

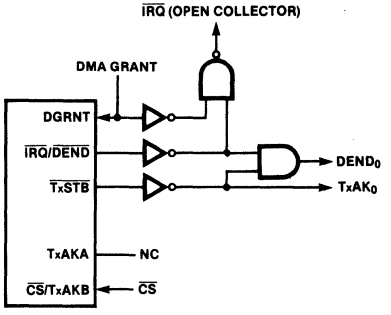


Fig. 7 Two-channel System

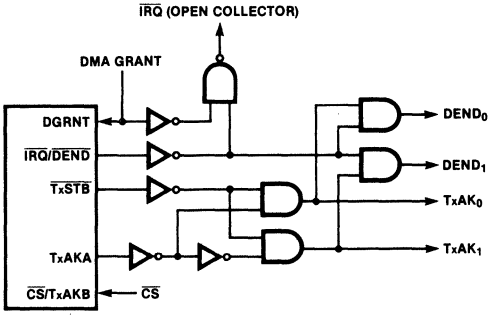
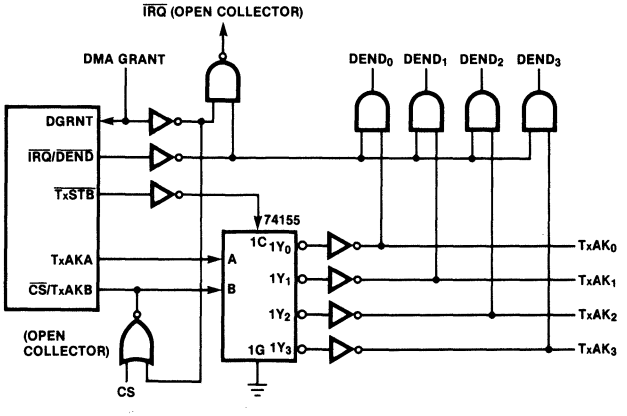


Fig. 8 Four-channel System



The four-channel circuit is functionally similar to the two-channel circuit but uses a 74LS139 to decode TxAKA and TxAKB for channel identification. Because the DMAC $\overline{CS}$/TxAKB pin is bidirectional during four-channel operation, an open-collector gate must be used to drive $\overline{CS}$ to avoid drive contention.

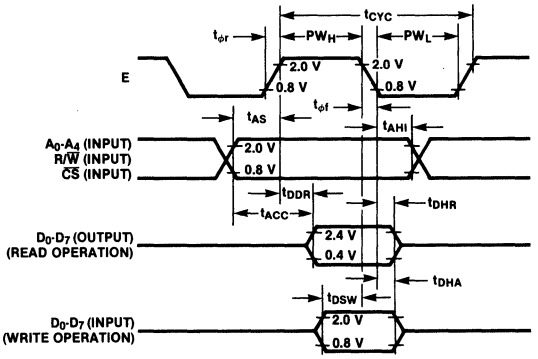
Timing Characteristics

The bus timing characteristics of the F6844 are described in *Table 4* and illustrated in *Figure 9*. The DMA timing characteristics are presented in *Table 5*. (Refer to *Figures 10* through *15* for the associated timing diagrams.) *Figure 16* illustrates the test loads and *Figure 17* the $\overline{CS}$/TxAKB source current test circuit.

Table 4 Bus Timing Characteristics

Symbol	Characteristic	Min	Max	Unit
Read Timing				
t_{AS}	Address Setup Time A_0-A_4, R/$\overline{W}$, $\overline{CS}$	160		ns
t_{AHI}	Address Input Hold Time A_0-A_4, R/$\overline{W}$, $\overline{CS}$	10		ns
t_{DDR}	Data Delay Time D_0-D_7		320	ns
t_{ACC}	Data Access Time D_0-D_7		480	ns
t_{DHR}	Data Output Hold Time D_0-D_7	10		ns
Write Timing				
t_{AS}	Address Setup Time A_0-A_4, R/$\overline{W}$, $\overline{CS}$	160		ns
t_{AHI}	Address Input Hold Time A_0-A_4, R/$\overline{W}$, $\overline{CS}$	10		ns
t_{DSW}	Data Setup Time D_0-D_7	195		ns
t_{DHW}	Data Input Hold Time D_0-D_7	10		ns
E Clock Timing				
t_{cyc}	Cycle Time	1000		ns
PWE_H	Pulse Width — High	450		ns
PWE_L	Pulse Width — Low	430		ns
$t\phi r$, $t\phi f$	Rise and Fall Time		25	ns

Fig. 9 Bus Timing



5

Table 5 DMA Timing Characteristics

Symbol	Characteristic	Min	Max	Unit
t_{TQS1}	TxRQ Setup Time, E Rising Edge	120		ns
t_{TQS2}	TxRQ Setup Time, E Falling Edge	210		ns
t_{TQH1}	TxRQ Hold Time, E Rising Edge	20		ns
t_{TQH2}	TxRQ Hold Time, E Falling Edge	20		ns
t_{DGS}	DGRNT Setup Time	155		ns
t_{DGH}	DGRNT Hold Time	10		ns
t_{AD}	Address Output Delay Time A_0-A_{15}, R/$\overline{W}$, $\overline{TxSTB}$		270	ns
t_{AHO}	Address Output Hold Time A_0-A_{15}, R/$\overline{W}$, $\overline{TxSTB}$	30 35		ns
t_{ATSD}	Address Three-State Delay Time A_0-A_{15}, R/$\overline{W}$		270	ns
t_{ATSR}	Address Three-State Recovery Time		270	ns
t_{DQD}	Delay Time, $\overline{DRQH}$, $\overline{DRQT}$		375	ns
t_{TKD1}	TxAK Delay Time, E Rising Edge		400	ns
t_{TKD2}	TxAK Delay Time, DGRNT Rising Edge		190	ns
t_{DED1}	$\overline{IRQ}$/$\overline{DEND}$ Delay Time, E Falling Edge		300	ns
t_{DED2}	$\overline{IRQ}$/$\overline{DEND}$ Delay Time, DGRNT Rising Edge		190	ns

Fig. 10 TxRQ Input Timing

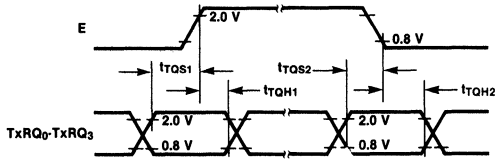
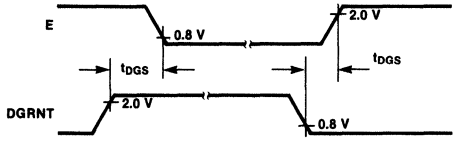


Fig. 11 DGRNT Input Timing

Setup Timing



Hold Timing

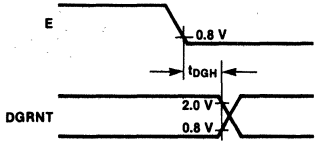


Fig. 12 $\overline{DRQH}$, $\overline{DRQT}$, TxAK Output Timing

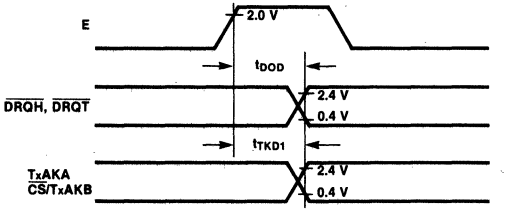


Fig. 13 Address, $\overline{IRQ}$/$\overline{DEND}$ Output Timing

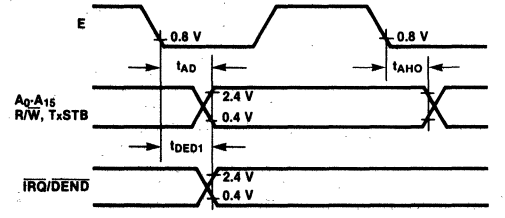


Fig. 14 Address Three-state Timing

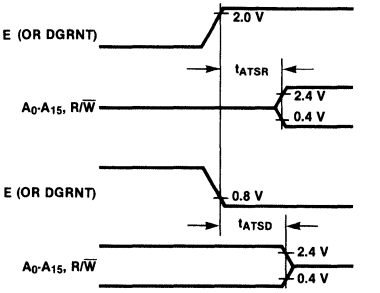


Fig. 15 TxAKB, $\overline{IRQ}$/$\overline{DEND}$ Output Timing from DGRNT Input

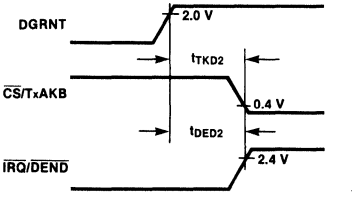
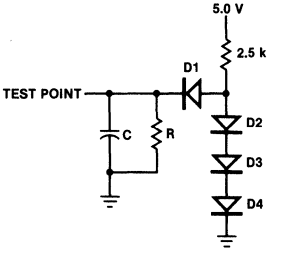
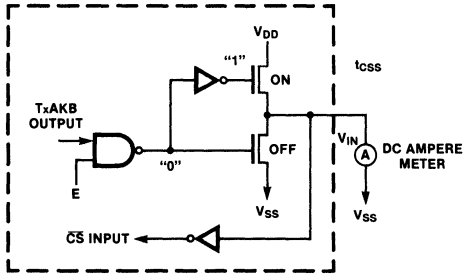


Fig. 16 Test Loads



Test Pin	C = pF	R = kΩ
D_0-D_7	130	11.7
A_0-A_{15}, R/$\overline{W}$	90	16.5
$\overline{CS}$/TxAKB	50	24
Others	30	24

Fig. 17 $\overline{CS}$/TxAKB Source Current Test Circuit



Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Storage Temperature	−55°C, +150°C
Operating Temperature	0°C, +70°C
Supply Voltage (V_{DD})*	−0.3 V, +7.0 V
Input Voltage (V_{IN})*	−0.3 V, +7.0 V
Thermal Resistance	70°C/W

*With respect to V_{SS}.

DC Characteristics

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
V_{IH}	Input High Voltage	$V_{SS}+2.0$		V_{DD}	V	
V_{IL}	Input Low Voltage	$V_{SS}-0.3$		$V_{SS}+0.8$	V	
I_{IN}	Input Leakage Current $TxRQ_0$-$TxRQ_3$, $\phi 2$ DMA, $\overline{RES}$, DGRNT			2.5	μA	$V_{IN}=0$ to 5.25 V
I_{TSI}	3-State Leakage Current A_0-A_{15}, R/$\overline{W}$, D_0-D_7	−10		10	μA	$V_{IN}=0.4$ to 2.4 V
V_{OH}	Output High Voltage D_0-D_7 A_0-A_{15}, R/$\overline{W}$ All Others	 $V_{SS}+2.4$ $V_{SS}+2.4$ $V_{SS}+2.4$			V	 $I_L=-205$ μA $I_L=-145$ μA $I_L=-100$ μA
V_{OL}	Output Low Voltage			$V_{SS}+0.4$	V	$I_L=1.6$ mA
I_{CSS}	Source Current $\overline{CS}$/TxAKB		10	16	mA	$V_{IN}=0$ V
P_D	Power Dissipation		500	1000	mW	
C_{IN}	Input Capacitance E D_0-D_7, $\overline{CS}$/TxAKB, A_0-A_4, R/$\overline{W}$ All Others			 20 12.5 10	 pF	 $V_{IN}=0$ V, $T_A=25$°C f = 1.0 MHz
C_{OUT}	Output Capacitance			12	pF	

$V_{DD}=5.0$ V ± 5%, $V_{SS}=0$ V, $T_A=0$°C to +70°C, unless otherwise noted.

Ordering Information

Order Code	Temperature Range
F6844P	0°C to +70°C
F68A44P	0°C to +70°C
F68B44P	0°C to +70°C

FAIRCHILD
A Schlumberger Company

F6845/F6845A
CRT Controller

Microprocessor Product

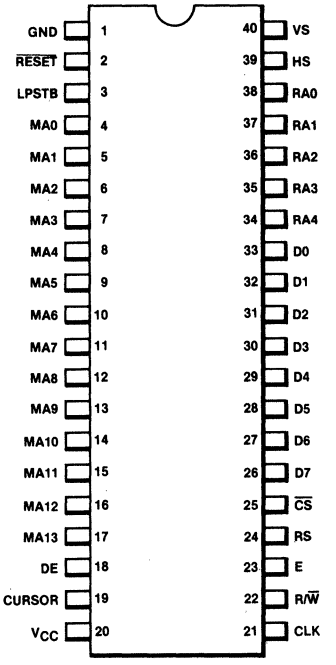
Description

The Fairchild F6845 CRT Controller (CRTC) provides an interface between a microprocessor (MPU) and a raster scan CRT device. The CRTC is used in microprocessor-based controller systems for CRT terminals in stand-alone or multiterminal configurations, including smart, programmable CRT terminals, video games, and information displays.

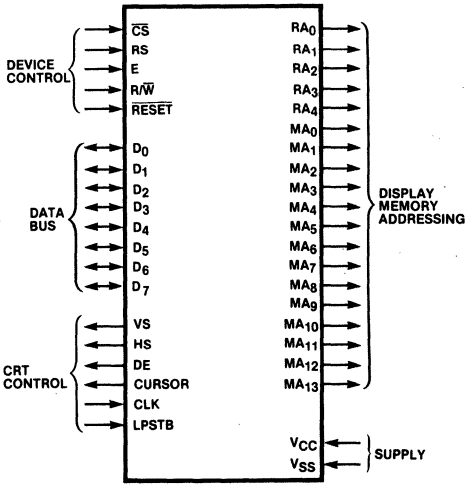
The F6845 CRTC is designed with an optimum hardware/software balance that achieves integration of all key functions and maintains flexibility. All keyboard functions, read/write operations, cursor movements, and editing are under microprocessor control. The F6845 provides video timing and refresh memory addressing.

- **Monochrome or Color CRT Applications**
- **Used with "Glass-Teletype", Smart, Programmable, Intelligent CRT Terminals; Video Games; Information Displays**
- **Alphanumeric, Semigraphic, and Full Graphic Capability**
- **Fully Programmable via Processor Data Bus; Timing can be Generated for Almost Any Alphanumeric Screen Format (e.g., 80 x 24, 72 x 64, and 132 x 20)**
- **Single +5 V Supply**
- **F6800-Compatible Bus Interface**
- **TTL-Compatible Inputs and Outputs**
- **Start Address Register Provides Hardware Scroll (By Page, Line, or Character)**
- **Programmable Cursor Register Allows Control of Cursor Format and Blink Rate**
- **Light Pen Register**
- **Refresh (Screen) Memory Can Be Multiplexed Between the CRTC and the MPU, Thus Removing the Requirements for Line Buffers or External DMA Devices**
- **Programmable Interlace or Non-Interlace Scan Modes**
- **14-Bit Refresh Address Allows up to 16K of Refresh Memory for Use in Character or Semigraphic Displays**
- **5-Bit Row Address Allows up to 32 Scan-Line Character Blocks**
- **512K Address Space is Available for Graphics System by Using Both the Refresh and Row Addresses**
- **Refresh Addresses are Provided During Retrace, Allowing the CRTC to Provide Row Addresses to Refresh Dynamic RAM**
- **Programmable Skew for Cursor and Display Enable (DE)**

Connection Diagram



Signal Functions



5

Figure 1 Non-Interlace Raster Scan System

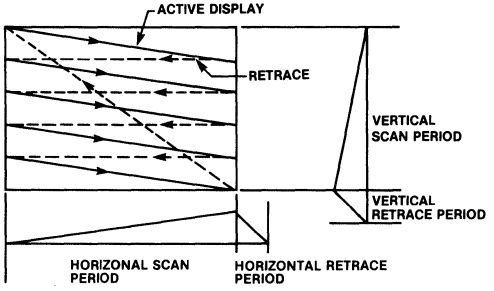
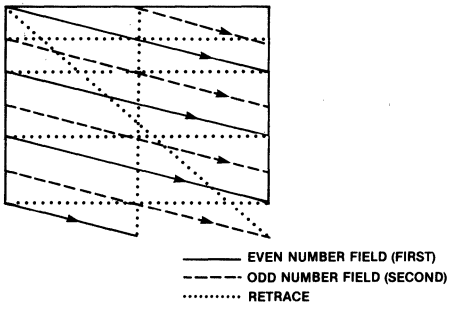


Figure 2 Interlace Raster Scan System



CRTC System Interface

The CRTC generates the signals necessary to interface a digital system to a raster scan CRT display. In this type of display, an electron beam starts in the upper left-hand corner, moves quickly across the screen, and returns. This action is called a horizontal scan. After each horizontal scan, the beam is incrementally moved down in the vertical direction until it has reached the bottom of the screen. At this point, one frame has been displayed, as the beam has made many horizontal scans and one vertical scan.

Two types of raster scanning are used in CRTs: interlace and non-interlace (illustrated in figures 1 and 2). Non-interlace scanning consists of one field per frame. The scan lines in figure 1 are shown as solid lines, and the retrace patterns are indicated by the dotted lines. Increasing the number of frames per second decreases the flicker. Ordinarily, either a 50 or 60 frame-per-second refresh rate is used to minimize beating between the CRT and the power line frequency. This prevents the displayed data from weaving.

Interlace scanning is used in broadcast TV and on data monitors where high-density or high-resolution data must be displayed. Two fields, or vertical scans, are made down the screen for each single picture or frame. The first field (even field) starts in the upper left-hand corner; the second (odd field) in the upper center. Both fields overlap as shown in figure 2, thus interlacing the two fields into a single frame.

To display the characters on the CRT screen, the frames must be continually repeated. The data to be displayed is stored in the refresh (screen) memory by the MPU controlling the data processing system. The data is usually written in ASCII code and cannot be directly displayed as characters. A character generator ROM is typically used to convert the ASCII codes into the "dot" pattern for every character.

The most common method of generating characters is to create a matrix of dots, x dots (columns) wide and y dots (rows) high. Each character is created by selectively filling in the dots. As x and y get larger, a more detailed character can be created. Two common dot matrices are 5 x 7 and 7 x 9. Many variations of these standards allow Chinese, Japanese, or Arabic letters instead of English. Since characters require some space between them, a character block larger than the character is typically used, as shown in figure 3. The figure also shows the corresponding timing and levels for a video signal that would generate the characters.

The CRTC generates the refresh addresses ($MA_0 - MA_{13}$), row addresses ($RA_0 - RA_4$), and the video timing — vertical synchronization (VS), horizontal synchronization (HS), and display enable (DE), as illustrated in figure 4. Other functions include an internal cursor register that generates a cursor output when its contents compare to the current refresh address. A light pen strobe input signal allows capture of the refresh address in an internal light pen register.

All timing in the CRTC is derived from the clock (CLK) input. In alphanumeric terminals, this signal is the character rate. The video rate, or dot clock, is externally divided by high-speed logic (TTL) to generate the CLK input. The high-speed logic also generates the timing and control signals necessary for the shift register, latch, and multiplexer control.

Figure 3 Character Display on the Screen and Video Signal

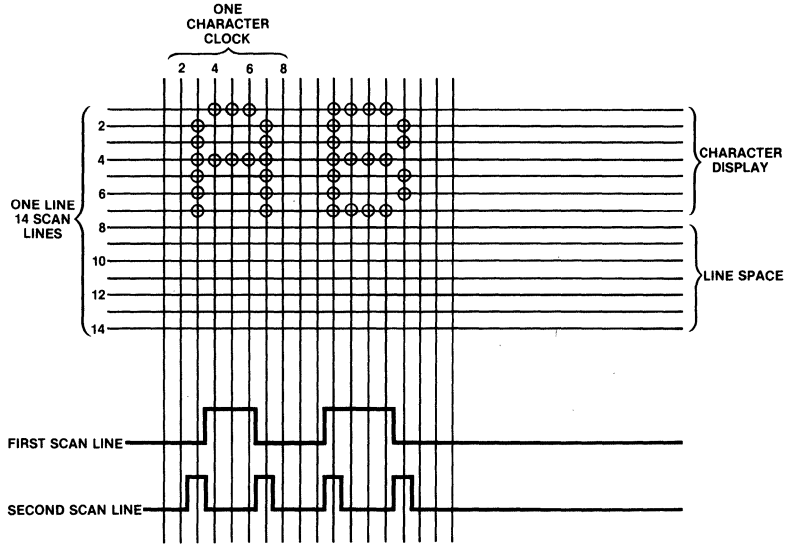
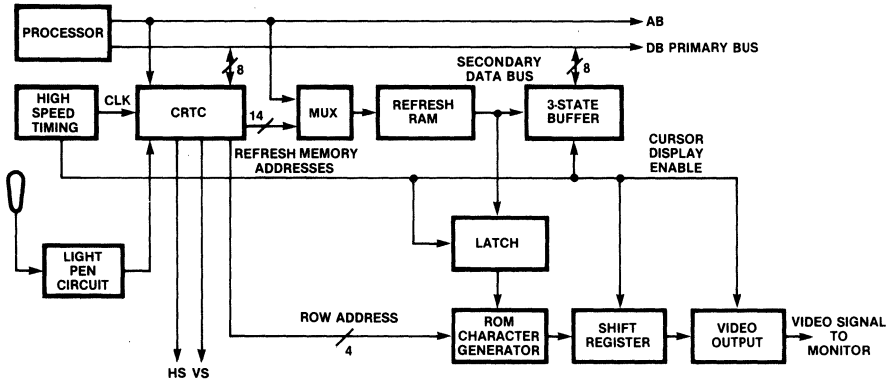


Figure 4 Typical CRT Controller Application



The processor communicates with the CRTC through an 8-bit data bus by reading or writing into the 19 registers. The refresh memory address is multiplexed between the microprocessor and the CRTC. Data appears on a secondary bus separate from the processor's primary bus. The secondary data bus concept in no way precludes using the refresh RAM for other purposes. It looks like any other RAM to the processor.

Refresh Memory Contentions

A number of approaches are possible for solving contentions in the refresh memory.

1. The processor always has priority. Generally, "hash" occurs, as the MPU and CRTC clocks are not synchronized.
2. The processor has priority access anytime, but can be synchronized by an interrupt to perform accesses only during horizontal and vertical retrace times.
3. The processor is synchronized with the memory wait cycles (states).
4. The processor is synchronized to the character rate as shown in figure 5. The F6800 processor family works very well in this configuration, as constant cycle lengths are present. This method provides no overhead for the processor, as there is never a contention for a memory access. All accesses are transparent.

The CRTC is offered in two pin-compatible versions. This data sheet contains information describing both the F6845 CRTC and the F6845A (upgraded) CRTC. Complete software compatibility between both versions is maintained by programming all register bits in the F6845A, which are undefined/unused in the F6845, with zeros.

The F6845 CRTC consists of programmable horizontal and vertical timing generators, programmable linear address register, programmable cursor logic, light pen capture register, and control circuitry for interface to a processor bus. Figure 6 is a functional block diagram of the CRTC.

All CRTC timing is derived from the clock (CLK) input, usually the output of an external dot rate counter. Coincidence (CO) circuits continuously compare the contents of the horizontal, horizontal sync width, character row, and scan line counters to the contents of the programmable register file, R_0 - R_{17}. For horizontal timing generation, comparisons result in the horizontal sync pulse (HS) of a frequency, position, and width determined by the registers, and the horizontal display signal of a frequency, position, and duration determined by the registers.

The horizontal counter produces H clock, which drives the scan line counter and vertical control. The contents of the scan line register raster counter are continuously compared to the contents of the scan line address register. A coincidence resets the raster scan line counter and clocks the vertical counter.

Comparisons of vertical counter contents and vertical registers result in a vertical sync (VS) pulse of a frequency, width, and position determined by the registers, and a vertical display of a frequency and position determined by the registers. The width of the VSYNC pulse is fixed at 16 raster lines in the F6845. The vertical control logic has other functions, including the following.

1. Generates row selects or raster address RA_0 - RA_4 output from the raster count scan line for the corresponding interlace or non-interlace modes.
2. Extends the number of scan lines in the vertical total by the amount programmed in the vertical total adjust register.

The linear address generator is driven by CLK and associates the relative positions of characters in memory with their positions on the CRT screen. Fourteen refresh memory address lines, $MA_0 - MA_{13}$, are available for addressing up to four pages of 4K characters, eight pages of 2 K characters, etc. Using the start address register, hardware scrolling up to 16K characters is possible. The linear address generator repeats the same sequence of addresses for each scan line of a character row.

The cursor logic determines the cursor location, size, and blink rate on the CRT screen. All are programmable.

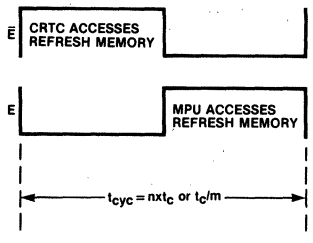
The light pen strobe (LPSTB) going high causes the current contents of the address counter to be latched in the light pen register. The contents of the light pen register are subsequently read by the microprocessor.

Internal CRTC registers are programmed by the microprocessor through the data bus, $D_0 - D_7$, and the control signals R/$\overline{W}$, $\overline{CS}$, RS, and E.

Signal Descriptions

The F6845 input/output signals are described in table 1.

Figure 5 Transparent Memory Configuration Timing Using F6800 MPU



Where m, n are integers and t_c is character period.

Figure 6 F6845 CRTC Block Diagram

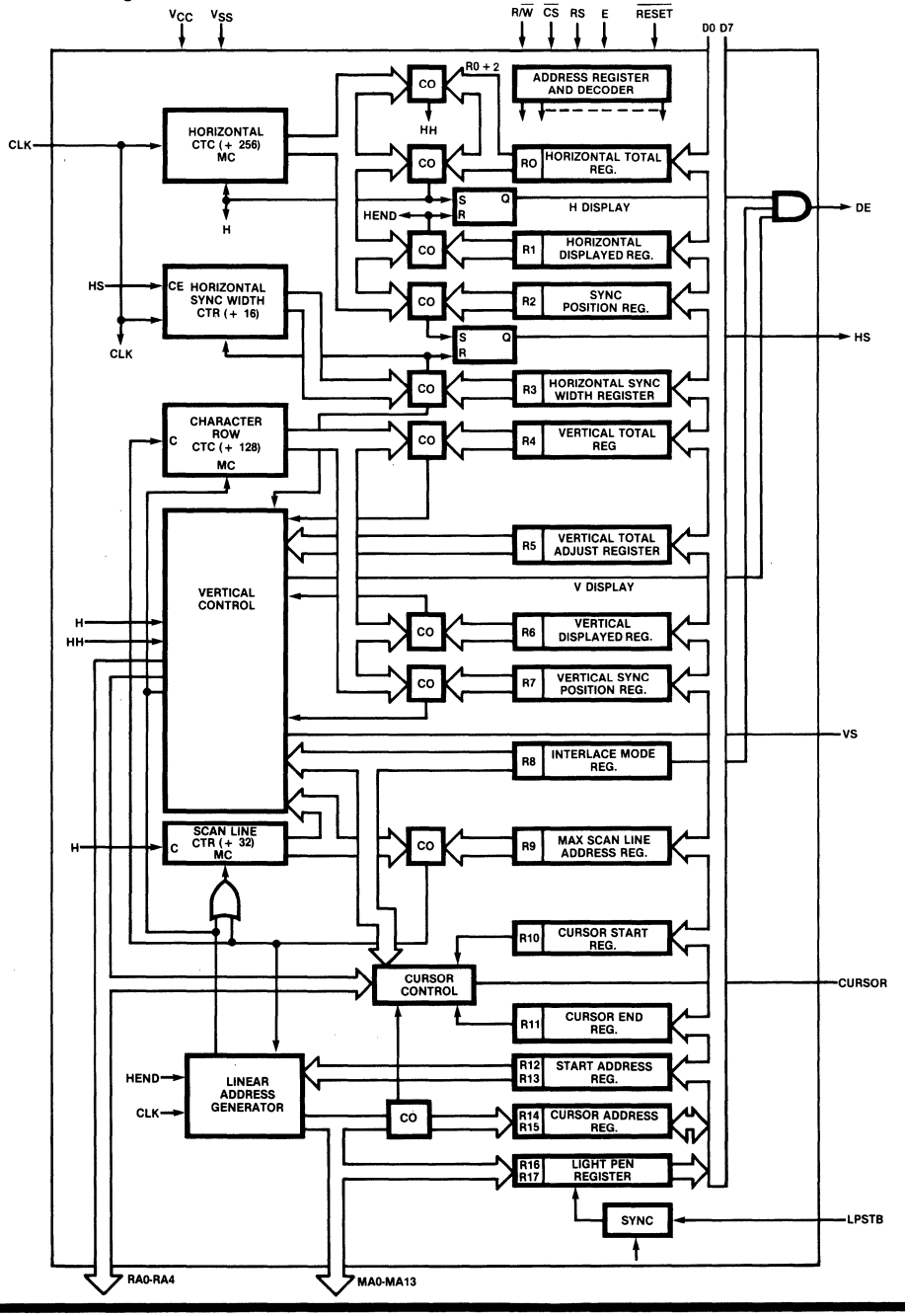


Table 1 F6845 CRTC Signal Descriptions

Mnemonic	Pin No.	Name	Description
Device Control			
$\overline{CS}$	25	Chip Select	High-impedance, TTL/MOS compatible input signal. When low, it selects the CRTC to allow reading from or writing to the internal register file. This signal should be active only when a valid stable address from the microprocessor is being decoded.
RS	24	Register Select	High-impedance, TTL/MOS compatible input signal. When low, it selects the address register; when high, it selects one of the data registers of the internal register file.
E	23	Enable	High-impedance, TTL/MOS compatible input signal. Enables the data bus input/output buffers and clocks data to and from the CRTC. This signal is usually derived from the microprocessor clock; the high-to-low transition is the active edge.
R/$\overline{W}$	22	Read/Write	High-impedance, TTL/MOS compatible input signal. Determines whether the internal register file is written to or read from. A write operation is an active low (logic 0).
$\overline{RESET}$	2	Reset	An input signal used to reset the CRTC. When low, it clears all CRTC counters, stops display operations, and forces all outputs low; control registers in the CRTC are not affected and remain unchanged. The $\overline{RESET}$ signal performs a reset function only when the $\overline{LPSTB}$ signal is also low, as shown in the following. $\overline{RESET}$ — $\overline{LPSTB}$ — Operating Mode 0 — 0 — Reset 0 — 1 — Test 1 — 0 — Normal 1 — 1 — Normal The test mode configures the memory addresses as two independent 7-bit counters to minimize test time. After $\overline{RESET}$ goes low, the $MA_0 - MA_{13}$ and $RA_0 - RA_4$ signals, in synchronization with the CLK low level, also go low (at least one CLK signal is necessary for reset). The CRTC starts the display operation immediately after the release of the $\overline{RESET}$ signal. The DE signal is not active until after the first VS signal pulse occurs.

Mnemonic	Pin No.	Name	Description
Data Bus			
$D_0 - D_7$	26-33	Data Bus	Eight bidirectional data lines that allow data transfers between the CRTC internal register file and the microprocessor. The data bus output drivers are 3-state buffers that remain in the high-impedance state except when the microprocessor performs a CRTC read operation.
CRT Control			
VS	40	Vertical Synchronization	Active-high, TTL compatible output signal that determines the vertical position of the displayed text, drives the monitor directly, or is fed to video processing logic for composite generation.
HS	39	Horizontal Synchronization	Active-high, TTL compatible output signal that determines the horizontal position of the displayed text, drives the monitor directly or is fed to video processing logic for composite generation.
DE	18	Display Enable	Active-high, TTL compatible output signal that indicates the CRTC is providing addressing in the active display area.
CURSOR	19	Cursor	Active-high, TTL compatible output signal that indicates valid cursor address to external video processing logic.
CLK	21	Clock	A TTL/MOS compatible input signal that is used to synchronize all CRT control signals, except for the processor interface. An external dot counter is used to derive this signal, which is usually the character rate in an alphanumeric CRT. The active transition is from high to low.
LPSTB	3	Light Pen Strobe	High-impedance, TTL/MOS compatible input signal that latches the current refresh addresses into the light pen register file. Latching occurs on the low-to-high edge, and is internally synchronized to the character clock.
Display Memory Addressing			
$MA_0 - MA_{13}$	4-17	Refresh Memory Addresses	Output signals that are used to refresh the CRT screen with pages of data located within a 16K block of refresh memory. Drives a standard TTL load and 30 pF.
$RA_0 - RA_4$	34-38	Row Addresses	Output signals from the internal row address counter that address the character ROM for the row of a character. These signals drive a standard TTL load and 30 pF.
Supply			
V_{CC}	20	Supply Voltage	+5 V supply.
V_{SS}	1	Ground	Supply and signal ground.

Register Files

The 19 registers of the F6845 CRTC can be accessed through the data bus. Only two memory locations are required, as one location is used as a pointer to address one of the remaining 18 registers. These 18 registers control horizontal timing, vertical timing, interlace operation, and row address operation, and define the cursor, cursor address, start address, and light pen register. The register addresses and sizes are shown in table 2.

Table 2 CRTC Internal Register Assignment

$\overline{CS}$	RS	Address Register 4	3	2	1	0	Register #	Register File	Program Unit	Read	Write	Number of Bits 7	6	5	4	3	2	1	0
1	X	X	X	X	X	X	X	—	—	—	—								
0	0	X	X	X	X	X	AR	Address Register	—	No	Yes								
0	1	0	0	0	0	0	R0	Horizontal Total	Char.	No	Yes								
0	1	0	0	0	0	1	R1	Horizontal Displayed	Char.	No	Yes								
0	1	0	0	0	1	0	R2	Sync Position	Char.	No	Yes								
0	1	0	0	0	1	1	R3	Sync Width	—	No	Yes	V_A	V_A	V_A	V_A	H	H	H	H
0	1	0	0	1	0	0	R4	Vertical Total	Char.Row	No	Yes								
0	1	0	0	1	0	1	R5	Total Adjust	Scan Line	No	Yes								
0	1	0	0	1	1	0	R6	Vertical Displayed	Char.Row	No	Yes								
0	1	0	0	1	1	1	R7	Sync Position	Char.Row	No	Yes								
0	1	0	1	0	0	0	R8	Interlace Mode & Skew	Note 1	No	Yes	C_A	C_A	D_A	D_A			I	I
0	1	0	1	0	0	1	R9	Max.Scan Line Address	Scan Line	No	Yes								
0	1	0	1	0	1	0	R10	Cursor Start	Scan Line	No	Yes		B	P		(Note 2)			
0	1	0	1	0	1	1	R11	Cursor End	Scan Line	No	Yes								
0	1	0	1	1	0	0	R12	Start Address (H)	—	Yes	Yes	0	0						
0	1	0	1	1	0	1	R13	Start Address (L)	—	Yes	Yes								
0	1	0	1	1	1	0	R14	Cursor (H)	—	Yes	Yes	0	0						
0	1	0	1	1	1	1	R15	Cursor (L)	—	Yes	Yes								
0	1	1	0	0	0	0	R16	Light Pen (H)	—	Yes	No	0	0						
0	1	1	0	0	0	1	R17	Light Pen (L)	—	Yes	No								

Notes

1. The skew control and interlace are described in the Interlace Mode and Skew Register section.
2. Bit 5 of the cursor start raster register is used for blink period control, and bit 6 is used to select blink or nonblink.
3. Subscript A represents the F6845A CRTC.

Address Register
The address register is a 5-bit write-only register used as an indirect or pointer register. It contains the address of one of the other 18 registers. When both the RS and $\overline{CS}$ signals are low, the address register is selected. When $\overline{CS}$ is low and RS is high, the register pointed to by the address register is selected.

Timing Registers R0-R9
The visible display area of a typical CRT monitor is shown in figure 7. The point of reference for horizontal registers is given as the left-most displayed character position. Horizontal registers are programmed in character clock time units with respect to the reference, as shown in figure 8. Signal characteristics are given in the "Timing Characteristics" section. The point of reference for the vertical registers is the top character position displayed. Vertical registers are programmed in scan line times with respect to the reference, as shown in figure 9.

Figure 7 F6845 CRT Screen Format

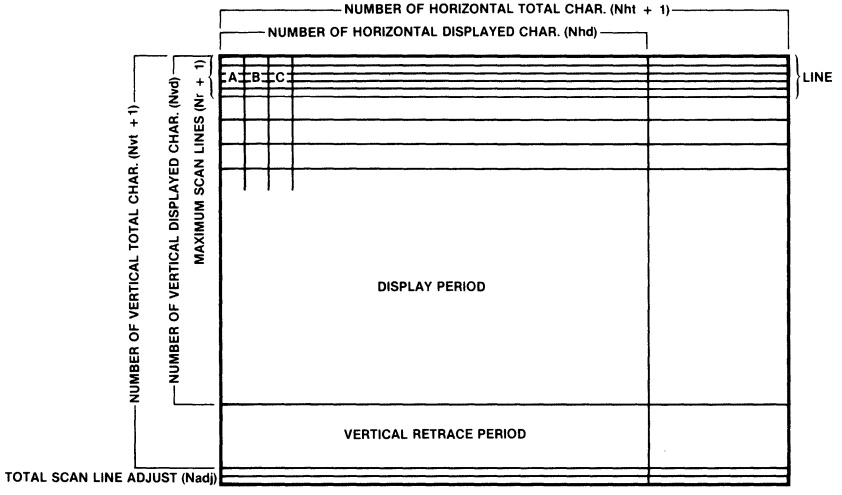
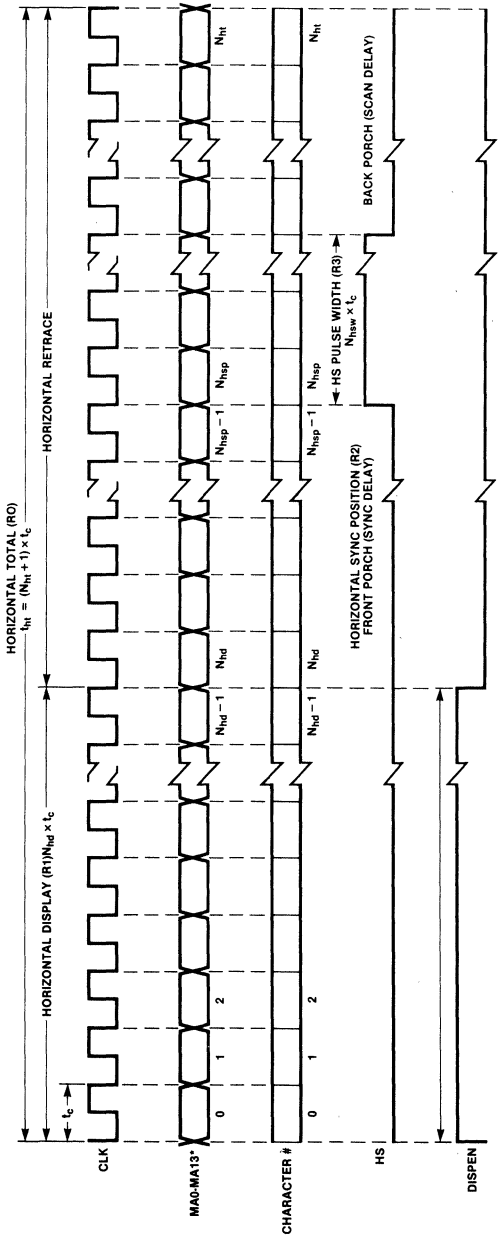
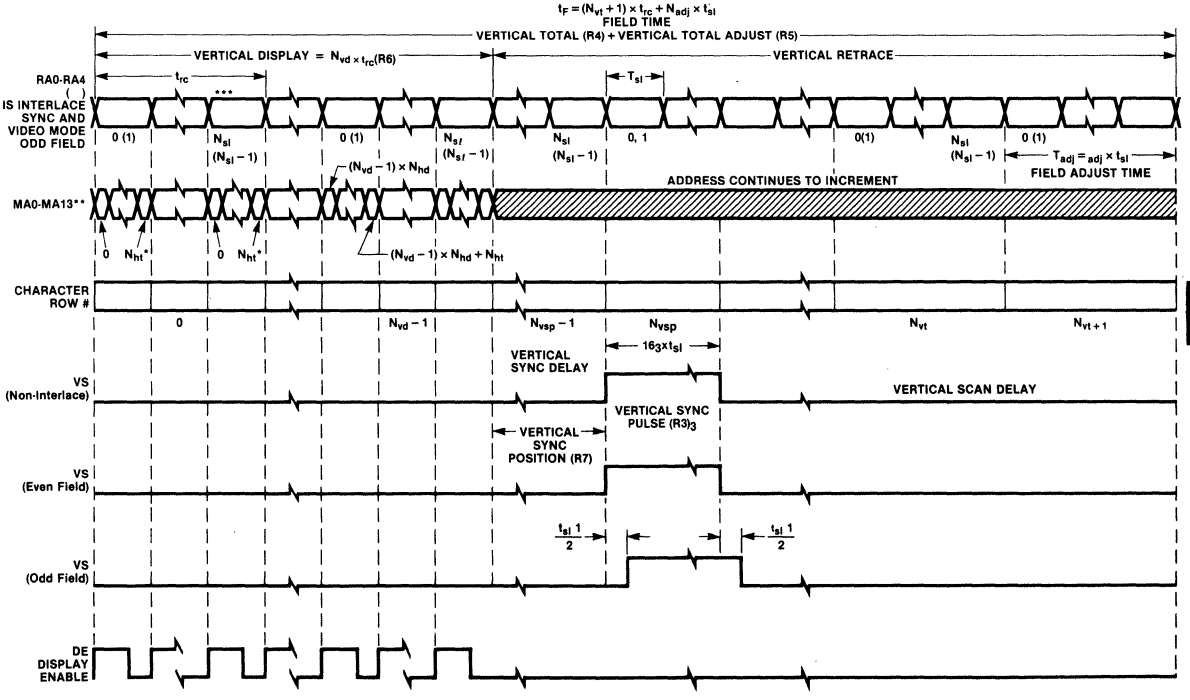


Figure 8 CRTC Horizontal Timing



Note
Timing is shown for first displayed scan row only. The initial MA is determined by the contents of start address register R_{12}/R_{13}. Timing is shown for $R_{12}/R_{13} = 0$.

Figure 9 CRTC Vertical Timing



*N_{ht} must be an odd number for both interlace modes.
**Initial MA is determined by R12/R13 (start address register), which is zero in this timing example.
***N_{sl} must be an odd number for interlace sync and video mode.

Notes:

1. Refer to Figure 2 - The odd field is offset ½ horizontal scan time.
2. Timing values are described in Table 9.
3. Vertical sync pulse width can be programmed from 1 to 16 scan line times for the MC6845 ☆ 1.

Horizontal Total Register (R0) - This 8-bit write-only register determines the horizontal sync (HSYNC) frequency by defining the period in character times. It is the total of the displayed characters plus the nondisplayed character times (retrace) minus one.

Horizontal Displayed Register (R1) - This 8-bit write-only register determines the number of displayed characters per line. Any 8-bit number can be programmed so long as the contents of R0 are greater than the contents of R1.

Horizontal Sync Position Register (R2) - This 8-bit write-only register controls the horizontal sync position, which defines the horizontal sync delay (Front Porch) and the horizontal scan delay (Back Porch). When the programmed value of this register is increased, the display on the CRT screen is shifted to the left. When the programmed value is decreased, the display is shifted to the right. Any 8-bit number can be programmed if the sum of the contents of R1, R2, and R3 is less than the contents of R0.

Sync Width Register (R3) - This 8-bit write-only register determines the width of the vertical sync pulse and the horizontal sync pulse for the F6845A CRTC. The vertical sync pulse width is fixed at 16 scan line times for the F6845, and the upper four bits of this register are treated as "don't cares".

The F6845A allows control of the VS pulse width for one to sixteen scan line times. Programming the upper four bits for one to fifteen selects pulse widths from one to fifteen scan line times. Programming the upper four bits as zeros selects a VS pulse width of 16 scan line times, allowing compatibility with the F6845.

For both the F6845 and the F6845A, the HS pulse width can be programmed from one to fifteen character clock periods, thus allowing compatibility with the HS pulse width specifications of many different monitors. If zero is written into this register, then no horizontal sync is provided.

This horizontal width must be programmed because, were it fixed as an integral of character times, it would vary with the character rate and be out of tolerance for certain monitors.

Horizontal Timing Summary - The difference between R0 and R1 is the horizontal blanking interval (refer to figure 8). This interval in the horizontal scan period allows the beam to return (retrace) to the left side of the screen. The retrace time is determined by the monitor's horizontal scan components. Retrace time is less than the horizontal blanking interval.

A good rule of thumb is to make the horizontal blanking about 20% of the total horizontal scanning period for a CRT. In inexpensive TV receivers, the beam overscans the display screen so that aging of parts does not result in underscanning. Because of this, the retrace time should be about one-third the horizontal scanning period. The horizontal sync delay, HS pulse width, and horizontal scan delay are typically programmed with a l:2:2 ratio.

Vertical Total Register (R4) and Vertical Total Adjust Register (R5) - The frequency of the VS pulse is determined by both the R4 and R5 registers. The calculated number of character line times is usually an integer plus a fraction to get exactly a 50 or 60 Hz vertical refresh rate. The integer number of character line times minus one is programmed in the 7-bit write-only vertical total register (R4). The fraction of character line times is programmed in the 5-bit write-only vertical total adjust register (R5) as a number of scan line times.

Vertical Displayed Register (R6) - This 7-bit write-only register specifies the number of displayed character rows on the CRT screen and is programmed in character row times. Any number smaller than the contents of the R4 register can be programmed into the R6 register.

Vertical Sync Position Register (R7) - This 7-bit write-only register controls the position of vertical sync with respect to the reference. It is programmed in character row times. The value programmed in the register is one less than the number of computed character line times. When the programmed value of this register is increased, the display position of the CRT screen is shifted up. When the programmed value is decreased, the display position is shifted down. Any number equal to or less than the vertical total (register R4) can be used.

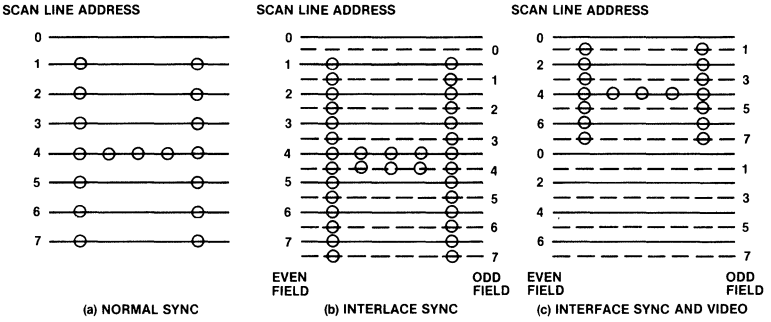
Table 3 Cursor and DE Skew Control

Value	Skew
00	No Character Skew
01	One Character Skew
10	Two Character Skew
11	Not Available

Table 4 Interlace Mode Register

Bit 1	Bit 0	Mode
0	0	Normal Sync Mode (Non-Interlace)
1	0	Normal Sync Mode (Non-Interlace)
0	1	Interlace Sync Mode
1	1	Interlace Sync and Video Mode

Figure 10 Interlace Control



Interlace Mode and Skew Register (R8) - The F6845 only allows control of the interlace modes as programmed by the low order two bits of this write-only register. The F6845A controls the interlace modes and allows a programmable delay of zero to two character clock times for the display enable (DE) and cursor outputs. Table 3 describes operation of the cursor and DE skew bits. Cursor skew is controlled by bits 6 and 7 of Register R8, while DE skew is controlled by bits 4 and 5. Table 4 shows the available interlace modes; these modes are selected using the two low order bits of this 6-bit write-only register. In the normal sync mode (non-interlace), only one field is available, as shown in figures 1 and 10 (a). Each scan line is refreshed at the VSYNC frequency (e.g., 50 or 60 Hz).

Two interlace modes are available, as shown in figures 2, 10 (b) and 10 (c). The frame time is divided between even and odd alternating fields. The horizontal and vertical timing relationship VSYNC delayed by one-half scan line time) results in the displacement of scan lines in the odd field with respect to the even field.

In the interlace sync mode, the same information is painted in both fields, as shown in figure 10 (b). This is a useful mode for filling in a character to enhance readability.

In the interlace sync and video mode, shown in figure 10 (c), alternating lines of the character are displayed in the even field and the odd field. This effectively doubles the given bandwidth of the CRT monitor.

To avoid an apparent flicker effect, care must be taken when using either interlace mode. This flicker effect is due to the doubling of the refresh time for all scan lines, since each field is displayed alaernately and can be minimized with proper monitor design (e.g., longer persistence phosphors).

In addition, the programming of the CRTC registers for interlace operation has the following restrictions.

F6845 Programming Restrictions

1. The horizontal total register value, R0, must be odd (i.e., an even number of character times).

2. For interlace sync and video mode only, the maximum scan line address, R9, must be odd (i.e., an even number of scan lines).

3. For interlace sync and video mode only, the vertical displayed register, R6, must be even. The programmed number, Nvd, must be one-half the actual number required. The even-numbered scan lines are displayed in the even field and the odd-numbered scan lines are displayed in the odd field.

4. For interlace sync and video mode only, the cursor start register, R10, and cursor end register, R11, must both be even or odd, depending in which field the cursor is to be displayed.

F6845A Programming Restrictions

1. The horizontal total register value, R0, must be odd (i.e., an even number of character times).

2. For the interlace sync and video mode only, the vertical displayed register, R6, must be even. The programmed number, Nvd, must be one-half the actual number required.

Maximum Scan Line Address Register (R9) - This 5-bit write-only register determines the number of scan lines per character row, including the spacing, thus controlling operation of the row address counter. The programmed value is a maximum address and is one less than the number of scan lines.

Figure 11 Cursor Control

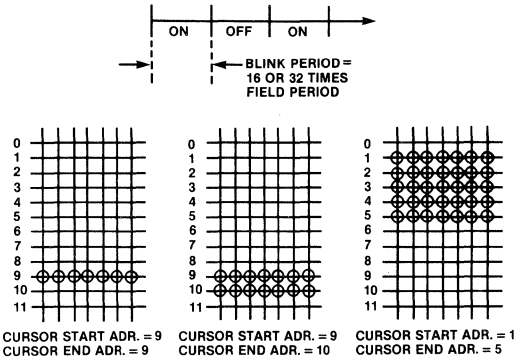


Table 5 Cursor Start Register

Bit 6	Bit 5	Cursor Display Mode
0	0	Non-blink
0	1	Cursor Non-display
1	0	Blink, 1/16 Field Rate
1	1	Blink, 1/32 Field Rate

Example of Cursor Display Mode

Cursor Control Registers-Cursor movement is controlled by the following four registers.

Cursor Start Register (R10) and Cursor End Register (R11) - These registers allow a cursor of up to 32 scan lines in height to be placed on any scan line of the character block, as shown in figure 11. Register R10 is a 7-bit write-only register used to define the start scan line and the cursor blink rate. Bits 5 and 6 of the cursor start address register control the cursor operation, as shown in table 5. Non-display, display, and two blink modes (16 times or 32 times the field period) are available. Register R11 is a 5-bit write-only register that defines the last scan line of the cursor.

Bit 5 is the blink timing control; when it is low, the blink frequency is 1/16 of the vertical field rate, and when it is high, the blink frequency is 1/32 of the vertical field rate. Bit 6 is used to enable a blink. The cursor start scan line is set by the lower five bits.

When an external blink feature on characters is required, it may be necessary to perform cursor blink externally so that both blink rates are synchronized. Note that an invert/noninvert cursor is easily implemented by programming the CRTC for a blinking cursor and externally inverting the video signal with an exclusive-OR gate.

Cursor Register (R14-H, R15-L) — This 14-bit read/write register pair is programmed to position the cursor anywhere in the refresh RAM area, thus allowing hardware paging and scrolling through memory without loss of the original cursor position. It consists of an 8-bit low order ($MA_0 - MA_7$) register and a 6-bit high order ($MA_8 - MA_{13}$) register.

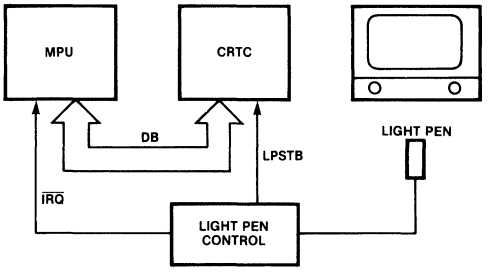
Start Address and Light Pen Registers

The following 14-bit registers control the start address and light pen.

Start Address Register (R12-H, R13-L) — This 14-bit write-only register pair controls the first address by the CRTC after vertical blanking. It consists of an 8-bit low order ($MA_0 - MA_7$) register and a 6-bit high order ($MA_8 - MA_{13}$) register. The start address register determines which portion of the refresh RAM is displayed on the CRT screen. Because the CRTC linear address generator counts from this beginning count, the displayed portion of the screen may be a window on any continuous string of characters within a 16 block of refresh memory. Hardware scrolling by characters, line, or page can be accomplished by centering the R12/R13 pointer in the middle of the available memory space.

Light Pen Register (R16-H, R17-L) - This 14-bit read-only register pair captures the refresh address output by the CRTC on the positive edge of a pulse input to the LPSTB pin. It consists of an 8-bit low order (MA_0-MA_7) register and a 6-bit high order (MA_0-MA_{13}) register. Since the light pen pulse is aynchronous with respect to refresh address timing, an internal synchronizer is designed into the CRTC. Due to delays in this circuit, the value of R16 and R17 need to be corrected in software. (See the bus timing diagram in the Timing Characteristics section). Figure 12 shows an interrupt-driven approach, although a polling routine could be used.

Figure 12 Light Pen Interface



CRTC Initialization

Registers R0-R15 must be initialized after the system power is turned on. The processor normally loads the CRTC register file sequentially from a firmware table, after which, in most systems, R0-R11 are not changed. The worksheet of table 6 is useful in computing proper register values for the CRTC. Table 6 shows the worksheet completed for an 80 × 24 configuration using a 7 × 9 character generator, and figure 13 shows an F6800 program that could be used to program the CRT controller. The programmed values allow use of either an F6845 or an F6845A CRTC.

The CRTC registers have an initial value at power up. When using a direct drive monitor (without horizontal oscillator), these initial values can result in out-of-tolerance operation. The CTRC programming should be done immediately after power up, especially in this type of system.

CRT Interface Signal Timing

Timing charts of CRT interface signals are illustrated with the aid of a programmed example of the CRTC. When values listed in table 7 are programmed into the CRTC control registers, the device provides the outputs as shown in the timing diagrams (figure 8, 9, 14, and 15). The screen format of this example is shown in figure 7, which illustrates the relation between refresh memory address (MA_0-MA_{13}), row address (RA_0-RA_4), and the position on the screen. In this example, the start address is assumed to be zero.

The bus timing test load is shown in figure 16; figure 17 illustrates the CRTC timing, and figure 18 illustrates the CRTC clock, memory addressing, and light pen timing. All signal timing characteristics are given in the "Timing Characteristics" section of this data sheet.

Additional CRTC Applications

The foremost system function that can be performed by the CRTC is the refreshing of dynamic RAM. This is quite simple, as the refresh addresses run continually.

Note that the LPSTB input signal can be used to support additional system function other than a light pen. A digital-to-analog converter (DAC) and comparator could be configured to use the refresh addresses as a reference to a DAC composed of a resistive adder network connected to a comparator. The output of the comparator generates the LPSTB input signal, signifying a match between the refresh address analog level and the unknown voltage.

The light pen strobe input could also be used as a character strobe to allow the CRTC refresh addresses to decode a keyboard matrix. Debouncing would need to be done in software.

Both the VS and HS signal outputs can be used as a real-time clock. Once programmed, the CRTC provides a stable reference frequency.

Table 6 Worksheet for 80 x 24 Format

Display Format Worksheet		
1. Displayed Characters per Row	80	Char.
2. Displayed Character Rows per Screen	24	Rows
3. Character Matrix a. Columns	7	Columns
b. Rows	9	Rows
4. Character Block a. Columns	9	Columns
b. Rows	11	Rows
5. Frame Refresh Rate	60	H_z
6. Horizontal Oscillator Frequency	16,600	H_z
7. Active Scan Lines (Line 2 × Line 4b)	264	Lines
8. Total Scan Lines (Line 6 ÷ Line 5)	310	Lines
9. Total Rows Per Screen (Line 8 ÷ Line 4b)	28	Rows and 2 Lines
10. Vertical Sync Delay (Character Rows)		Rows
11. Vertical Sync Width (Scan Lines, 16)	16	Lines
12. Horizontal Sync Delay (Character Times)	6	Character Times
13. Horizontal Sync Width (Character Times)	9	Character Times
14. Horizontal Scan Delay (Character Times)	7	Character Times
15. Total Character Times (Lines 1 + 12 + 13 + 14)	102	Character Times
16. Character Rate (Line 6 times 15)	1.8972 M	MH_z
17. Dot Clock Rate (Line 4a times 16)	17.075 M	MH_z

CRTC Registers

		Decimal	Hex
R0	Horizontal Total (Line 15 minus 1)	101	65
R1	Horizontal Displayed (Line 1)	80	50
R2	Horizontal Sync Position (Line 1 + Line 12)	86	56
R3	Horizontal Sync Width (Line 13)	9	9
R4	Vertical Total (Line 9 minus 1)	24	18
R5	Vertical Adjust (Line 9 Lines)	10	0A
R6	Vertical Displayed (Line 2)	24	18
R7	Vertical Sync Position (Line 2 + Line 10)	24	18
R8	Interlace (00 Normal, 01 Interlace 03 Interlace, and Vidio)		0
R9	Max. Scan Line Add (Line 4b minus 1)	11	B
R10	Cursor Start	0	0
R11	Cursor End	11	B
R12, R13	Start Address (H and L)	128	00 80
R14, R15	Cursor (H and L)	128	00 80

Figure 13 F6800 Program for CRTC Initialization

Page 001 CRTC INIT. SA:0 F6845/F6845-1 CTRC initialization program

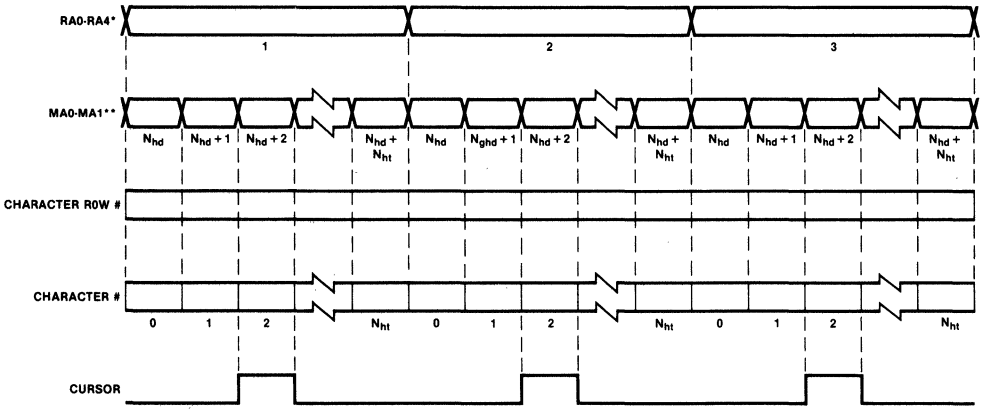
```
                                NAM    F6845
                                TTL    F6845-1     CRTC initialization program
                                OPT    G,S,LLE=85 print FCB's, FDB's & XREF table
                         ********************************************************
                         *Assign  CRTC   addresses
                         *
                         CRTCAD EQU    $9000       Address Register
                         CRTCRG EQU    CRTCAD+1    Data Register
                         ********************************************************
                         * Initialization program
                         *
A 0000                          ORG    0           a place to start
A 0000 5F                       CLRB               clear counter
A 0001 CE 1020 A                LDX    #CRTTAB     table pointer
A 0004 F7 9000 A         CRTC1  STAB   CRTCAD      load address register
A 0007 A6 00   A                LDAA   0,X         get register value from table
A 0009 B7 9001 A                STAA   CRTCRG      program register
A 000C 08                       INX                increment counters
A 000D 5C                       INCB
A 000E C1 10   A                CMPB   $10         finished?
A 0010 26 F2 0004               BNE    CRTC1       no: take branch
A 0012 3F                       SWI                yes: call monitor
                         ********************************************************
                         * CRTC register initialization table
                         *
A 1020                          ORG    $1020       start of table
A 1020   65    A         CRTTAB FCB    $65,$50     R0, R1 - total & H displayed
A 1021   50    A
A 1022   56    A                FCB    $56,$09     R2, R3 - pos. & HS width
A 1023   09    A
A 1024   18    A                FCB    $18,$0A     R4, R5 - V total & V total adj.
A 1025   0A    A
A 1026   18    A                FCB    $18,$18     R6, R7 - V displayed & VS pos.
A 1027   18    A
A 1028   00    A                FCB    $00,$0B     R8, R9 - Interlace & Max scan line
A 1029   0B    A
A 102A   00    A                FCB    $00,$0B     R10,R11 - Cursor start & end
A 102B   0B    A
A 102C   0080  A                FDB    $0080       R12,R13 - Start Address
A 102E   0080  A                FDB    $0080       R14,R15 - Cursor Address
                         END
Total Errors 00000--00000

CRTCL 0004 CRTCAD 9000 CRTCRG 9001 CRTTAB 1020
```

5

Table 7 Values Programmed into CRTC Registers

Reg.#	Register Name	Value	Programmed Value
R0	Horizontal Total	$N_{ht}+1$	N_{ht}
R1	Horizontal Displayed	N_{hd}	N_{hd}
R2	Horizontal Sync Position	N_{hsp}	N_{hsp}
R3	Horizontal Sync Width	N_{hsw}	N_{hsw}
R4	Vertical Total	$N_{vt}+1$	N_{vt}
R5	Vertical Scan Line Adjust	N_{adj}	N_{adj}
R6	Vertical Displayed	N_{vd}	N_{vd}
R7	Vertical Sync Position	N_{vsp}	N_{vsp}
R8	Interlace Mode		
R9	Max. Scan Line Address	N_{sl}	N_{sl}
R10	Cursor Start	1	
R11	Cursor End	3	
R12	Start Address (H)	0	
R13	Start Address (L)	0	
R14	Cursor (H)	0	
R15	Cursor (L)	2	
R16	Light Pen (H)		
R17	Light Pen (L)		

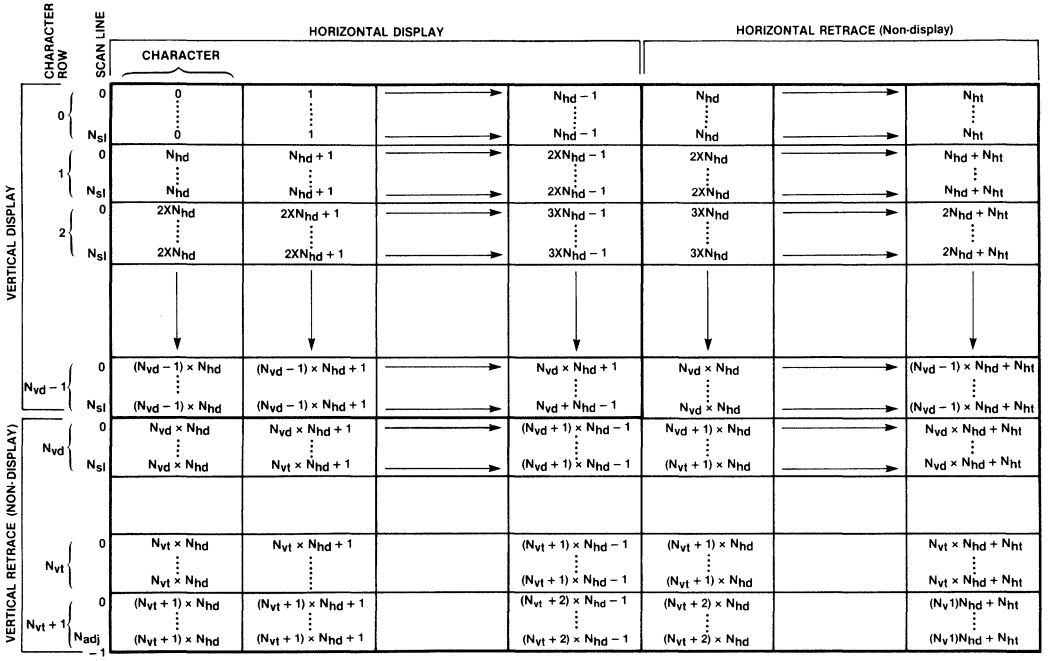
Figure 14 Cursor Timing Diagram



*Timing is shown for non-interlace and interlace sync modes.
Example shown has cursor programmed as
Cursor Register = $N_{hd}+2$
Cursor Start = 1
Cursor End = 3

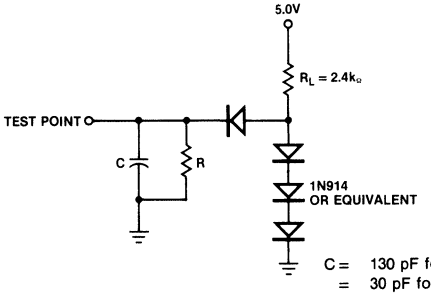
**The initial MA is determined by the contents of start address register, R12/R13. Timing is shown for R12/13 = 0

Figure 15 Refresh Memory Addressing ($MA_0 - MA_{13}$) Timing Diagram



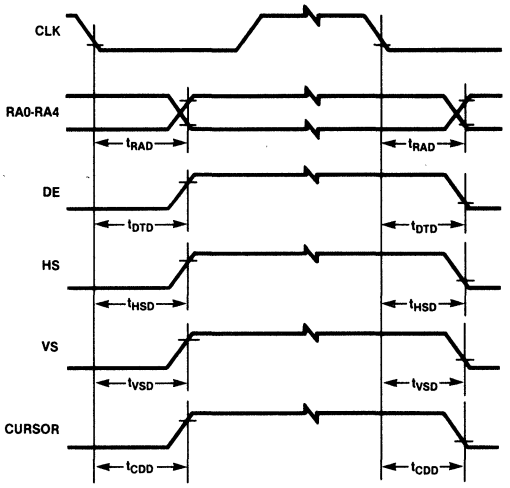
Note 1: The initial MA is determined by the contents of start register, R12/R13. Timing is shown for R12/R13=0. Only non-interlace and interlace sync modes are shown.

Figure 16 CRTC Bus Timing Test Load



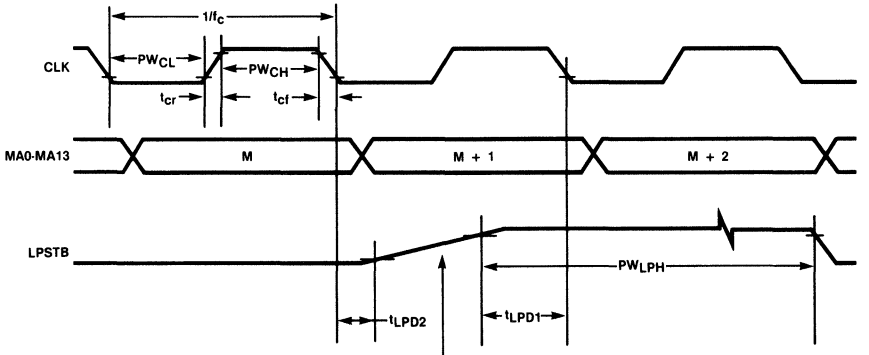
C = 130 pF for D0-D7
= 30 pF for MAO-MA13, RA0-RA4, DE, HS, VS, and CURSOR
R = 11 kΩ for D0-D7
= 24 for All Other Outputs

Figure 17 CRTC Timing Diagram



Note: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts unless otherwise noted.

Figure 18 CRTC Clock, Memory Addressing, and Light Pen Timing Diagram

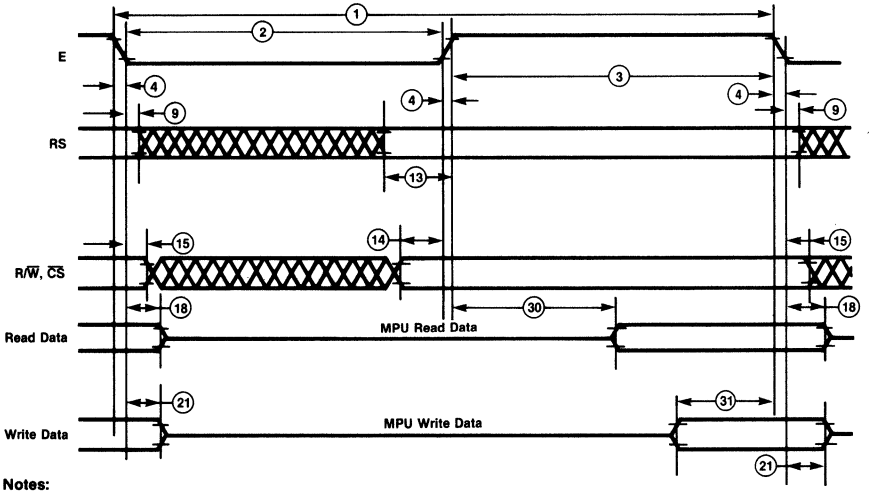


Note: Timing measurements are referenced to and from a low voltage of 0.8 volts, and a high voltage of 2.0 volts, unless otherwise noted.

Timing Characteristics

The signal timing for the CRTC bus is shown in figure *19*; ac characteristics for the bus timing are given in table *8*, and for the CRTC timing in table 9.

Figure 19 CRTC Bus Timing Diagram



Notes:
1. Voltage levels shown are $V_L \leq 0.4$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.

Table 8 CRTC Bus Timing Characteristics

Ident. Number	Characteristic	Symbol	F6845/ F6845A		F68A45/ F68A45A		F68B45/ F68B45A		Unit
			Min	Max	Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	10	0.67	10	0.5	10	μs
2	Pulse Width, E Low	PW_{EL}	430	9500	280	9500	210	9500	ns
3	Pulse Width, E High	PW_{EH}	450	9500	280	9500	20	500	ns
4	Clock Rise and Fall Time	t_r, t_f	—	25	—	25	—	20	ns
9	Address Hold Time (RS)	t_{AH}	10	—	10	—	10	—	ns
13	RS Setup Time Before E	t_{AS}	80	—	60	—	40	—	ns
14	R/W and CS Setup Time Before E	t_{CS}	80	—	60	—	40	—	ns
15	R/W and CS Hold Time	t_{CH}	10	—	10	—	10	—	ns
18	Read Data Hold Time	t_{DHR}	20	50•	20	50•	20	50•	ns
21	Write Data Hold Time	t_{DHW}	10	—	10	—	10	—	ns
30	Peripheral Output Data Delay Time	t_{DDR}	—	290	—	180	0	150	ns
31	Peripheral Input Data Setup Time	t_{DSW}	165	—	80	—	60	—	ns

• The data bus output buffers are no longer sourcing or sinking current by t_{DHR} max (high impedance).

Table 9 CRTC Timing Characteristics

Characteristic	Symbol	Min	Max	Unit
Minimum Clock Pulse Width, Low	P_{WCL}	160	—	ns
Minimum Clock Pulse Width, High	P_{WCH}	200	—	ns
Clock Frequency	f_c	—	2.5	MH_z
Rise and Fall for Clock Input	t_{cr}, t_{cf}	—	20	ns
Memory Address Delay Time	t_{MAD}	—	160	ns
Raster Address Delay Time	t_{RAD}	—	160	ns
Display Timing Delay Time	t_{DTD}	—	300	ns
Horizontal Sync Delay Time	t_{HSD}	—	300	ns
Vertical Sync Delay Time	t_{VSD}	—	300	ns
Cursor Display Timing Delay Time	t_{CDD}	—	300	ns
Light Pen Strobe Minimum Pulse Width	PW_{LPH}	100	—	ns
Light Pen Strobe Disable Time	t_{LPD1}	—	120	ns
	t_{LPD2}	—	0	ns

Note: The light pen strobe must fall to low level before the VSYNC pulse rises.

DC Characteristics

The DC characteristics for the F6845 CRTC are presented in table 10.

Table 10 CRTC DC Characteristics (V_{CC} = 5.0 V ± 5%, V_{SS} = 0°C to 70°C, unless otherwise noted.)

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
V_{IH}	Input High Voltage	2.0	—	V_{CC}	V	
V_{IL}	Input Low Voltage	−0.3	—	0.8		V
I_{IN}	Input Leakage Current	—	1.0	2.5	μA	
I_{TSI}	Three-State (V_{CC} = 5.25 V) (V_{IN} = 0.4 to 2.4 V)	−10	2.0	10	μA	
V_{OH}	Output High Voltage				V	
	(I_{load} = −205 μA)	2.4	—	—		D0-D7
	(I_{load} = −100 μA)	2.4	—	—		Other Outputs
V_{OL}	Output Low Voltage (I_{load} = 1.6 mA)	—	—	0.4	V	
P_D	Power Dissipation	—	600	—	mW	
C_{IN}	Input Capacitance				pF	
		—	—	12.5		D0-D7
		—	—	10		All others
C_{OUT}	Output Capacitance	—	—	10	pF	All Outputs

Absolute Maximum Ratings

Stresses greater than those indicated may cause permanent damage to the device. These stress ratings only, and functional operation of the F6845 under these or any other conditions above those indicated in this data sheet is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Supply Voltage	−0.3 V, +7.0 V
Input Voltage	−0.3 V, +7.0 V
Operating Temperature	0°C, +70°C
Storage Temperature	−55°C, +150°C

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6845P,S F6845AP,S	0°C to +70°C
	F6845CP,CS F6845ACP,CS	−40°C to +85°C
1.5 MHz	F68A45P,S F68A45AP,S	0°C to +70°C
	F68A45CP,CS F68A45ACP,CS	−40°C to +85°C
2.0 MHz	F68B45P,S F68B45AP,S	0°C to +70°C

FAIRCHILD
A Schlumberger Company

F6846
ROM-I/O-Timer

Microprocessor Product

Description

The F6846 combination chip provides the means, in conjunction with the F6802, to develop a basic 2-chip microcomputer system. The F6846 consists of 2048 bytes of mask-programmable Read Only Memory (ROM), an 8-bit bidirectional data port with control lines, and a 16-bit programmable timer-counter.

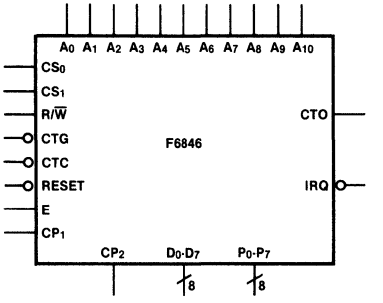
This device is capable of interfacing with the F6802 (basic F6800, clock and 128 bytes of RAM) as well as the F6800 if desired. No external logic is required to interface with most peripheral devices.

- **2048 8-Bit Bytes of Mask-Programmable ROM**
- **8-Bit Bidirectional Data Port for Parallel Interface Plus Two Control Lines**
- **Programmable Interval Timer-Counter Functions**
- **Programmable I/O Peripheral Data, Control, and Direction Registers**
- **Compatible with the Complete F6800 Microcomputer Product Family**
- **TTL-Compatible Data and Peripheral Lines**
- **Single 5 V Power Supply**

Pin Names

E	Enable (Clock System $\phi 2$) Input
A_0-A_{10}	Address Inputs
CS_0, CS_1	Chip Select Inputs
$R/\overline{W}$	Read/Write Input
$\overline{CTG}$	Counter Gate Input
$\overline{CTC}$	External Clock Input
$\overline{RESET}$	Reset Input
CP_1	Peripheral Interrupt Input
CP_2	Bidirectional Peripheral Control
D_0-D_7	Bidirectional Data Lines
P_0-P_7	Bidirectional Peripheral Data Lines
$\overline{CTO}$	Counter Timer Output
$\overline{IRQ}$	Interrupt Request Output

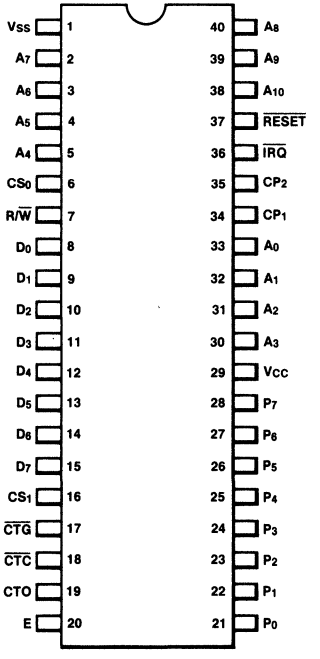
Logic Symbol



V_{CC} = Pin 29
V_{SS} = Pin 1

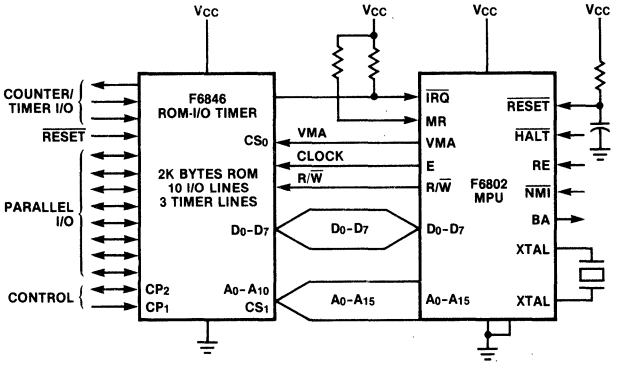
5

Connection Diagram
40-Pin DIP

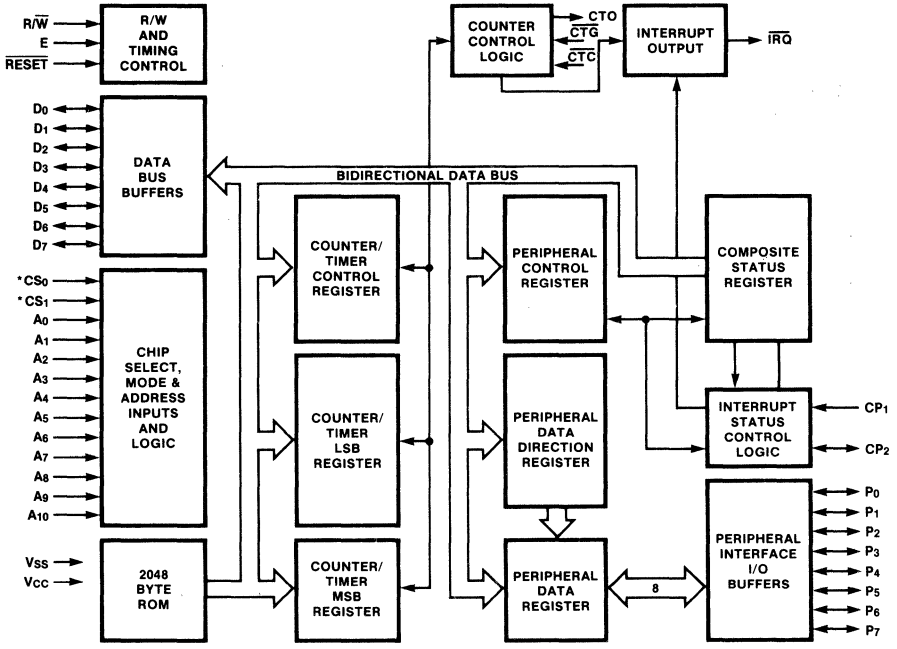


(Top View)

Typical Microcomputer



Block Diagram



*Mask Programmable

Functional Description

The F6846 combination chip may be partitioned into three functional operating sections: programmed storage, timer-counter functions, and a parallel I/O port.

Programmed Storage

The mask-programmable ROM section is similar to other ROM products of the F6800 family. The ROM is organized in a 2048 by 8-bit array to provide read-only storage for a minimum microcomputer system. Two mask-programmable chip selects are available for user definition.

Address inputs A_0–A_{10} allow any of the 2048 bytes of ROM to be uniquely addressed. Bidirectional data lines (D_0–D_7) allow the transfer of data between the MPU and the F6846.

Timer-Counter Functions

Under software control, this 16-bit binary counter may be programmed to count events, measure frequencies and time intervals, or similar tasks. It may also be used for square wave generation, single pulses of control duration and gated delayed signals. Interrupts may be generated from a number of conditions selectable by software programming.

The Timer/Counter Control Register allows control of the interrupt enable, output enable and selection of an internal or external clock source, a divide-by-eight prescaler, and operating mode. The Counter-Timer Clock ($\overline{CTC}$) will accept an asynchronous pulse to decrement the internal register for the counter-timer. If the divide-by-eight prescaler is used, the maximum clock rate can be four times the master clock frequency with an absolute maximum of 4 HMz. Counter Gate input ($\overline{CTG}$) accepts an asynchronous TTL-compatible signal that may be used as a trigger or gating function to the counter-timer. The Counter Timer Output (CTO) is also available and is under software control, being dependent on the timer control register, the gate input, and the clock source.

Parallel I/O Port

The parallel bidirectional I/O port has functional characteristics similar to the B port on the F6821 PIA. This includes eight bidirectional data lines and two handshake control signals. The control and operation of these lines are completely software programmable. Internal registers associated with the I/O functions may be selected with A_0, A_1 and A_2.

The Peripheral Interrupt input (CP_1) will set the interrupt flag (CSR_1) of the Composite Status Register. The Peripheral Control (CP_2) may be programmed to act as an interrupt input (set CSR_2) or as a Peripheral Control output.

Pin Functions

Bus Interface

The F6846 interfaces to the F6800 bus via an 8-bit bidirectional data bus, two chip select lines, a read/write line, and 11 address lines. These signals, in conjunction with the F6800 VMA Output, permit the MPU to control the F6846.

Bidirectional Data Bus (D_0-D_7)

The bidirectional data lines (D_0–D_7) allow the transfer of data between the MPU and the F6846. The data bus output drivers are 3-state devices that remain in the high-impedance (OFF) state except when the MPU performs an F6846 register or ROM read (R/$\overline{W}$ HIGH and I/O registers or ROM selected).

Chip Select (CS_0, CS_1)

The CS_0 and CS_1 inputs are used to select the ROM or I/O timer of the F6846. They are mask programmed to be active HIGH or active LOW as specified by the user.

Address (A_0-A_{10})

The Address inputs allow any of the 2048 bytes of ROM to be uniquely selected when the circuit is operating in the ROM mode. In the I/O-Timer mode, Address inputs A_0, A_1 and A_2 select the proper I/O register, while A_3 through A_{10} (together with CS_0 and CS_1) can be used as additional qualifiers in the I/O select circuitry. (See the section on I/O-Timer Select Circuitry for additional details.)

Reset ($\overline{RESET}$)

The active LOW state of the $\overline{RESET}$ input is used to initialize all register bits in the I/O section of the device to their proper values. (See the section on Initialization for timer and peripheral register reset conditions.)

Enable (E)

This signal synchronizes data transfer between the MPU and the F6846. It also performs an equivalent synchronization function on the External Clock, $\overline{RESET}$ and Counter Gate inputs of the F6846 timer section.

Read/Write (R/$\overline{W}$)

This signal is generated by the MPU and is used to control the direction of data transfer on the bidirectional data lines. A LOW level on the R/$\overline{W}$ input enables the F6846 input buffers and data is transferred to the circuit during the $\phi 2$ pulse when the part has been selected. A

HIGH level on the R/$\overline{W}$ input enables the output buffers and data is transferred to the MPU during $\phi 2$ when the part is selected.

Interrupt Request ($\overline{IRQ}$)
The active LOW $\overline{IRQ}$ output acts to interrupt the MPU through logic included on the F6846. This output utilizes an open drain configuration and permits interrupt request outputs from other circuits to be connected in a wire-OR configuration.

Peripheral Data (P_0-P_7)
The Peripheral Data lines can be individually programmed as either inputs or outputs via the Peripheral Data Direction Register. When programmed as outputs, these lines will drive two standard TTL loads (3.2 mA). They are also capable of sourcing up to 1.0 mA at 1.5 V (logic HIGH output).

When programmed as inputs, the output drivers associated with these lines enter a 3-state (high-impedance) mode. Since there is no internal pull-up for these lines, they represent a maximum 10 μA load to the circuitry driving them, regardless of logic state.

A logic LOW at the $\overline{RESET}$ input forces the Peripheral Data lines to the input configuration by clearing the Peripheral Data Direction Register. This allows the system designer to preclude the possibility of having a peripheral data output connected to an external driver output during power-up sequence.

Peripheral Interrupt (CP_1)
Peripheral Interrupt input CP_1 sets the interrupt flags of the Composite Status Register. The active transition for this signal is programmed by the Peripheral Control Register for the parallel port. CP_1 may also act as a strobe for the Peripheral Data Register when it is used as an input latch. Details for programming CP_1 are in the section on the parallel peripheral port.

Peripheral Control (CP_2)
Peripheral Control CP_2 may be programmed to act as an interrupt input or Peripheral Control output. As an input, this line has high impedance and is compatible with standard TTL voltage levels. As an output, it is TTL-compatible and may be used as a source of 1 mA at 1.5 V to directly drive the base of a Darlington transistor switch. CP_2 is programmed by the Peripheral Control Register.

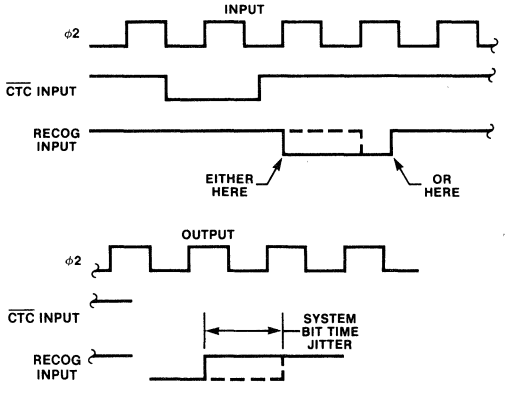
Counter Timer (CTO)
The Counter Timer output is software programmed by selected bits in the Counter/Timer Control Register. The mode of operation is dependent on the Counter/Timer Control Register, the Counter Gate input, and the clock source. The output is TTL compatible.

External Clock ($\overline{CTC}$)
$\overline{CTC}$ will accept asynchronous TTL voltage level signals to be used as a clock to decrement the timer. The HIGH and LOW levels of the external clock must be stable for at least one system clock period plus the sum of the set-up and hold times for the inputs. The asynchronous clock rate can vary from dc to the limit imposed by system $\phi 2$, set-up and hold times.

The External Clock input is clocked in by enable (system $\phi 2$) pulses. Three enable periods are used to synchronize and process the external clock. The fourth enable pulse decrements the internal counter. This does not affect the input frequency; it merely creates a delay between a clock input transition and internal recognition of that transition by the F6846. All references to $\overline{CTC}$ inputs in this document relate to internal recognition of the input transition. Note that a clock transition that does not meet set-up and hold-time specifications may require an additional enable pulse for recognition.

When observing recurring events, a lack of synchronization will result in either *system jitter* or *input jitter* being observed on the output of the F6846 when using an asynchronous clock and gate input signal. System jitter is the result of input signals out of synchronization with the system $\phi 2$ clock (Enable), permitting signals with marginal set-up and hold time to be recognized within either the bit-time nearest the input transition or subsequent bit-time.

Input jitter can be as great as the time between input signal negative-going transitions plus the system jitter if the first transition is recognized during one system cycle, and not recognized the next cycle or vice-versa.



Counter Gate ($\overline{CTG}$)

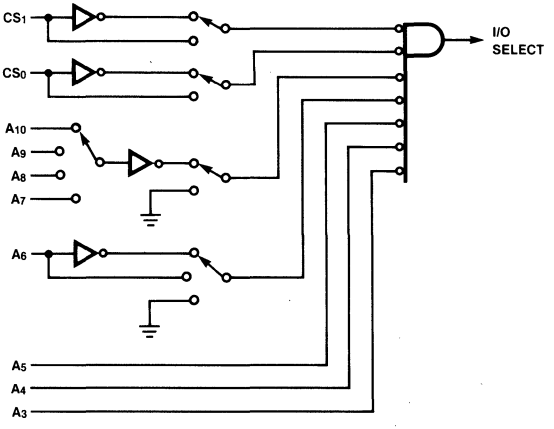
$\overline{CTG}$ accepts an asynchronous TTL-compatible signal which is used as a trigger or a clock-gating function to the timer. The gating input is clocked into the F6846 by the Enable (system $\phi 2$) signal in the same manner as the previously discussed clock inputs. A $\overline{CTG}$ transition is recognized on the fourth enable pulse, provided set-up and hold time requirements are met. The HIGH or LOW levels of the $\overline{CTG}$ input must be stable for at least one system clock period plus the sum of set-up and hold times. All references to $\overline{CTG}$ transition in this document relate to internal recognition of the input transition.

The $\overline{CTG}$ input of the timer directly affects the internal 16-bit counter. The operation of $\overline{CTG}$ is therefore independent of the divide-by-eight prescaler selection.

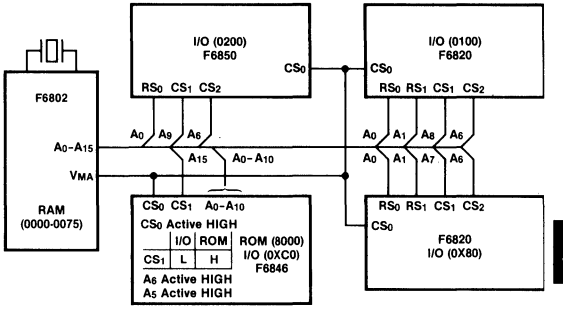
Functional Select Circuitry

I/O-Timer Select Circuitry

CS_0 and CS_1 are user programmable. Any of the four binary combinations of CS_0 and CS_1 can be used to select the ROM. Likewise, any other combination can be used to select the I/O-Timer. In addition, several Address lines are used as qualifiers for the I/O-Timer. Specifically, $A_3 = A_4 = A_5 =$ logic L. A_6 can be programmed to an H, L, or don't care (X). $A_7 = A_8 = A_9 = A_{10}$ = don't care or one line only may be programmed to a logic H. The available chip select options are diagramed below.



Memory mapping the I/O can be accomplished by using one of the CS inputs to select between ROM and I/O, applying the F6802 VMA output to the other CS (programmed active HIGH) and using the Address lines to decode Address fields.



5

Internal Addressing

Seven I/O register locations within the F6846 are accessible to the MPU data bus. Selection of these registers is controlled by A_0, A_1, and A_2 as shown in *Table 1*, provided the I/O timer is selected. CS_0 and CS_1 must be in the I/O state and the proper register address must be applied to access a particular register. The Composite Status Register is read only, where all other registers are read/write.

Table 1 Internal Register Addresses

Register Selected	A_2	A_1	A_0
Composite Status Register	L	L	L
Peripheral Control Register	L	L	H
Peripheral Data Direction Register	L	H	L
Peripheral Data Register	L	H	H
Composite Status Register	H	L	L
Counter/Timer Control Register	H	L	H
Counter/Timer MSB Register	H	H	L
Counter/Timer LSB Register	H	H	H
ROM Address	X	X	X

L = LOW Voltage Level
H = HIGH Voltage Level
X = Don't Care

Initialization

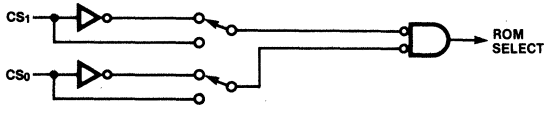
When the $\overline{\text{RESET}}$ input has accepted a LOW signal, all registers are initialized to the reset state. The Peripheral Data Direction and Peripheral Data Registers are cleared. The Peripheral Control Register is cleared except for bit 7 (the reset bit). This forces the parallel port to the input mode with interrupts disabled. To remove the reset condition from the parallel port, an L must be written into the Peripheral Control Register bit 7 (PCR_7).

During initialization the counter latches are preset to their maximal count, the Counter/Timer Control Register bits are reset to L except for bit 0 (TCR_0 is set), the counter output is cleared, and the counter clock disabled. This state forces the timer counter to remain in an inactive state. The Composite Status Register is cleared of all interrupt flags. During timer initialization the reset bit (TCR_0) must be cleared.

ROM Select Circuitry

The mask-programmable ROM section is similar in operation to other ROM products of the F6800 microprocessor family. The ROM is organized as 2048 words of 8-bits to provide read-only storage for a minimum microcomputer system. The ROM is active when selected by the unique combination of the chip select inputs.

The active levels of CS_0 and CS_1 for ROM and I/O select are a user programmable option. Either CS_0 and CS_1 may be programmed active HIGH or active LOW, but different codes must be used for ROM or I/O select. CS_0 and CS_1 are mask programmed simultaneously with the ROM pattern. The ROM select circuitry is shown below.



Timer Operation

The timer may be programmed to operate in modes which fit a wide variety of applications. The device is fully bus compatible with the F6800 system, and is accessed by load and store operations from the MPU.

In a typical application, the timer will be loaded by storing two bytes of data into the counter latch. This data is then transferred into the counter during a counter initialization cycle. The counter decrements on each subsequent clock cycle (which may be system $\phi 2$ or an external clock) until one of several predetermined conditions causes it to halt or recycle. Thus the timer is programmable, cyclic in nature, controllable by external inputs or MPU program, and accessible to the MPU at any time.

Counter Latch Initialization

The timer consists of a 16-bit addressable counter and two 8-bit addressable latches. The latches store a binary equivalent of the desired count value minus one. Counter initialization results in the transfer of the latch contents to the counter. It should be noted that data transfer to the counters is always accomplished via the latches. Thus, the counter latches may be accurately described as a 16-bit *counter initialization data* storage register.

In some modes of operation, the initialization of the latches will cause simultaneous counter initialization (i.e., immediate transfer of the new latch data into the counters). It is, therefore, necessary to insure that all 16 bits of the latches are updated simultaneously. Since the F6846 data bus is eight bits wide, a temporary register (MSB buffer register) is provided for the most significant byte of the desired latch data. This is a write-only register selected via Address lines A_0, A_1 and A_2. Data is transferred directly from the data bus to the MSB buffer when the chip is selected, R/$\overline{W}$ is LOW, and the timer MSB register is selected (A_0 = L; A_1 = A_2 = H).

The lower eight bits of the counter latch can also be referred to as a write-only register. Data bus information will be transferred directly to the LSB of a counter latch when the chip is selected, R/$\overline{W}$ is LOW and the Counter/Timer LSB Register is selected (A_0 = A_1 = A_2 = H). Data from the MSB buffer will be transferred automatically into the most significant byte of the counter latches simultaneously with the transfer of the data bus information to the least significant byte of the counter latch. For brevity, the conditions of this operation will be referred to henceforth as a *write-timer-latches command.*

The F6846 has been designed to allow transfer of two bytes of data into the counter latches from any source, provided that the MSB is transferred first. In many applications, the source of the data will be an F6800 MPU. It should therefore be noted that the 16-bit store operations of the F6800 family microprocessors (STS and STX) transfer data in the order required by the F6846. A store index register instruction, for example, results in the MSB of the X register being transferred to the selected address, then the LSB of the X register being written into the next higher location. Thus, either the index register or stack pointer contents may be transferred directly into either 8-bit latch with a single instruction.

A logic L at the $\overline{RESET}$ input also initializes the counter latches. All latches will assume maximum count (65,536) values. It is important to note that an internal reset (bit zero of the Counter/Timer Control Register set) has no effect on the counter latches.

Counter Initialization

Counter initialization is defined as the transfer of data from the latches to the counter with attendant clearing of the individual interrupt flag associated with the counter. Counter initialization always occurs when a reset condition (external $\overline{RESET}$ = L or TCR_0 = H) is recognized. It can also occur depending on the timer mode with a write-timer-latches command or recognition of a negative transition of the $\overline{CTG}$ input.

Counter recycling or reinitialization occurs when a clock input is recognized after the counter has reached an all L state. In this case, data is transferred from the latches to the counter, but the interrupt flag is unaffected.

Counter/Timer Control Register

The Counter/Timer Control Register (see *Table 2*) in the F6846 is used to modify timer operation to suit a variety of applications. The Counter/Timer Control Register has a unique address space (A_0 = H, A_1 = L, A_2 = H) and therefore may be written into at any time. The least significant bit of this control register is used as an internal reset bit. When this bit is a logic L, all timers are allowed to operate in the modes prescribed by the remaining bits of the control registers.

Writing H into Counter/Timer Control Register bit 0 (TCR_0) causes the counter to be preset with the contents of the counter latches, all counter clocks to be disabled, and the timer output and interrupt flag (status register) to be reset. The counter latch and Counter/Timer Control Register are undisturbed by an internal reset and may be written into regardless of the state of TCR_0.

Counter/Timer Control Register bit 1 (TCR_1) is used to select the clock source. When TCR_1 = L, the external clock input $\overline{CTC}$ is selected, and when TCR_1 = H, the timer uses system $\phi 2$.

Counter/Timer Control Register bit 2 (TCR_2) enables the divide-by-eight prescaler (TCR_2 = H). In this mode, the clock frequency is divided by eight before being applied to the counter. When TCR_2 = L the clock is applied directly to the counter.

TCR_3, TCR_4, and TCR_5 select the timer operating mode, and are discussed in the next section.

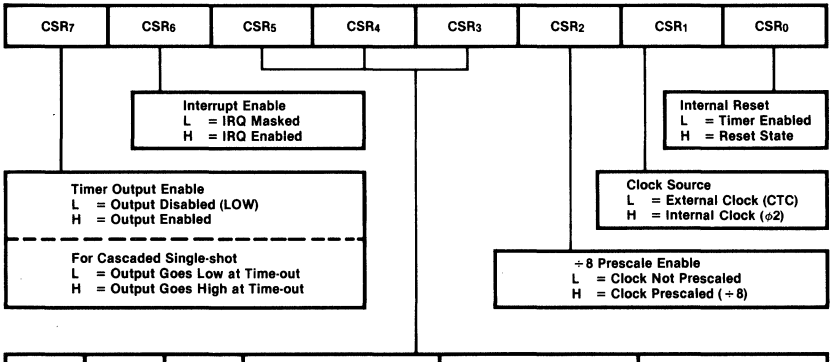
Counter/Timer Control Register bit 6 (TCR_6) is used to mask or enable the timer interrupt request. When TCR_6 = L, the interrupt flag is masked from the timer. When TCR_6 = H, the interrupt flag is enabled into bit 7 of the Composite Status Register (composite IRQ bit), which appears on the $\overline{IRQ}$ output.

Counter/Timer Control Register bit 7 (TCR_7) has a special function when the timer is in the cascaded single-shot mode. (This function is explained in detail in the section describing the mode.) In all other modes, TCR_7 merely acts as an output enable bit. If TCR_7 = L, the Counter Timer Output (CTO) is forced LOW. Writing a logic L into TCR_7 enables CTO.

Table 2 Counter/Timer Control Register Format

Control Register Bit	State	Bit Definition	State Definition
TCR_0	L	Internal Reset	Timer Enabled
	H		Timer in Preset State
TCR_1	L	Clock Source	Timer uses External Clock ($\overline{CTC}$)
	H		Timer uses $\phi 2$ System Clock
TCR_2	L	÷8 Prescaler Enabler	Clock is not Prescaled
	H		Clock is Prescaled by ÷8 Counter
TCR_3	X	Operating Mode Selection	See *Table 3*
TCR_4	X		
TCR_5	X		
TCR_6	L	Timer Interrupt Enable	$\overline{IRQ}$ Masked from Timer
	H		$\overline{IRQ}$ Enabled from Timer
TCR_7	L	Timer Output Enable	Counter Output (CTO) Set LOW
	H		Counter Output Enabled

Table 3 Counter/Timer Control Register



TCR_3	TCR_4	TCR_5	Timer Operating Mode	Counter Initialization	Interrupt Flag Set
L	L	L	Continuous	$\overline{CTG}$ ↓ + W + R	T.O.
L	L	H	Cascaded Single Shot	$\overline{CTG}$ ↓ + R	T.O.
L	H	L	Continuous	$\overline{CTG}$ ↓ + R	T.O.
L	H	H	Normal Single Shot	$\overline{CTG}$ ↓ + R	T.O.
H	L	L	Frequency Comparison	$\overline{CTG}$ ↓ • I • (W + T.O.) + R	$\overline{CTG}$ ↓ Before T.O.
H	L	H		$\overline{CTG}$ ↓ • I + R	T.O. Before $\overline{CTG}$ ↓
H	H	L	Pulse Width Comparison	$\overline{CTG}$ ↓ • I + R	$\overline{CTG}$ ↑ Before T.O.
H	H	H			T.O. Before $\overline{CTG}$ ↑

Timer Operating Modes

The F6846 has been designed to operate effectively in a wide variety of applications. This is accomplished by using three bits of the Counter/Timer Control Register (TCR_3, TCR_4 and TCR_5) to define different operating modes of the timer, outlined in *Table 4.*

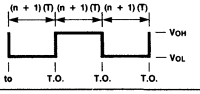
Table 4 Operating Modes

Control Register				
TCR_3	TCR_4	TCR_5	Timer Operating Mode	
L	*	L	Continuous	
L	L	H	Single-shot	Cascaded Single-shot
	H			Normal Single-shot
H	L	*	Frequency Comparison	
H	H	*	Pulse Width Comparison	

*Defines Additional Timer Functions

Continuous Operating Mode (TCR_3 = L, TCR_5 = L)
The timer may be programmed to operate in a continuous counting mode by writing L into bits 3 and 5 of the Counter/Timer Control Register. Assuming that the timer output is enabled (TCR_7 = H), a square wave will be generated at the Counter Timer Output CTO (see *Table 5*).

Table 5 Continuous Operating Modes (TCR_3 = L, TCR_7 = H, TCR_5 = L)

Control Register		Initialization/Output Waveforms	
TCR_2	TCR_4	Counter	Timer Output (2X)
L	L	Initialization $\overline{G}$ + W + R	
L	H	$\overline{G}$ + R	

$\overline{G}$ = Negative transition of $\overline{CTG}$ input
W = Write-timer-latches command
R = Timer Reset (TCR_0 = H or External $\overline{RESET}$ = L)
N = 16-bit number in Counter Latch
T = Period of Clock input to Counter
to = Counter initialization cycle
T.O. = Counter Time-out (all L condition)

Note
All time intervals shown above assume that the $\overline{CTG}$ and $\overline{CTC}$ signals are synchronized to system φ2 with the specified set-up and hold time requirements.

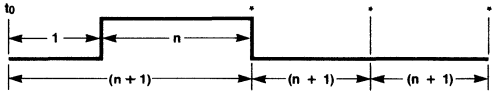
Either a Timer Reset (TCR_0 = H or External Reset = L) condition or internal recognition of a negative transition of the $\overline{CTG}$ input results in counter initialization. A write-timer-latches command can be selected as a counter initialization signal by clearing TCR_4.

The discussion of the continuous mode has assumed that the application requires an output signal. It should be noted that the timer operates in the same manner with the output disabled (TCR_7 = L). A read-timer-counter command is valid regardless of the state of TCR_7.

Single-shot Timer Mode
(TCR_3 = L, TCR_4 = H, TCR_5 = H)
This mode is identical to the continuous mode with two exceptions. The first of these is obvious from the name—the output returns to a LOW level after the initial time-out and remains LOW until another counter initialization cycle occurs. The internal counting mechanism remains cyclical in the single-shot mode. Each time-out of the counter results in the setting of an individual interrupt flag and reinitialization of the counter.

The second major difference between the single-shot and continuous modes is that the internal counter enable is not dependent on the $\overline{CTG}$ input level remaining in the LOW state for the single-shot mode.

Normal Single-Shot Mode Output Waveform



H = Write an "H" into TCR_7
L = Write an "L" into TCR_7

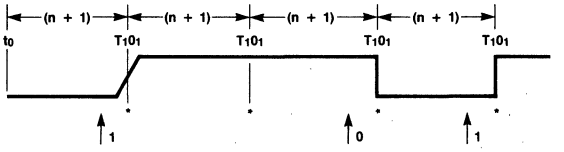
Note
All time intervals shown above assume the $\overline{CTG}$ and $\overline{CTC}$ signals are synchronized to system $\phi 2$ with the specified set-up and hold time requirements.

Cascaded Single-shot Mode
(TCR_3 = L, TCR_4 = L, TCR_5 = L)
This mode is identical to the single-shot mode with two exceptions. First, the output waveform does not return to LOW level and remain LOW after time-out. Instead, the output level remains at its initialized level until it is reprogrammed and changed by time-out. The output level may be changed at any time-out or may have any number of time-outs between changes.

The second difference is the method used to change the output level. Counter/Timer Control Register bit 7 (TCR_7) has a special function in this mode. The Counter Timer Output (CTO) is equal to TCR_7 clocked by time-out. At every time-out the content of TCR_7 is clocked to and held at the CTO. Thus, output pulses of length greater than one timer cycle can be generated by cascading timer cycles and counting time-outs with a software program.

An interrupt is generated at each time-out. To cascade timer cycles, the MPU would need an interrupt routine to: 1) count each time-out and determine when to change TCR_7; 2) write into TCR_7 the state corresponding to the next desired state of the output waveform (only necessary during the last timer cycle before the output is to change state); and 3) clear the interrupt flag by reading the Composite Status Register followed by the read timer MSB. It is also possible, if desired, to change the length of the timer cycle by reinitializing the timer latches. This allows more flexibility for obtaining desired times.

Cascaded Single-shot Mode Output Waveform



H = Write an "H" into TCR_7
L = Write an "L" into TCR_7

Note
All time intervals shown above assume the $\overline{CTG}$ and $\overline{CTC}$ signals are synchronized to system $\phi 2$ with the specified set-up and hold time requirements.

Time Interval Modes (TCR_3 = H)

The time interval modes are provided for applications requiring more flexibility of interrupt generation and counter initialization. The interrupt flag is set in these modes as a function of both counter time-out and $\overline{CTG}$ input transition. Counter initialization is also affected by interrupt flag status. The output signal is not defined in any of these modes. Other features of the time interval modes are outlined in *Table 6.*

Table 6 Timer Interval Modes (TCR_3 = H)

TCR_4	TCR_5	Application	Condition for Setting Individual Interrupt Flag
L	L	Frequency Comparison	Interrupt generated if $\overline{CTG}$ input period (1/F) is less than counter time-out (T.O.)
L	H	Frequency Comparison	Interrupt generated if $\overline{CTG}$ input period (1/F) is greater than counter time-out (T.O.)
H	L	Pulse Width Comparison	Interrupt generated if $\overline{CTG}$ input *down time* is less than counter time-out (T.O.)
H	H	Pulse Width Comparison	Interrupt generated if $\overline{CTG}$ input *down time* is greater than counter time-out (T.O.)

Frequency Comparison Mode (TCR_3 = H, TCR_4 = L)
The timer within the F6846 may be programmed to compare the period of a pulse (giving the frequency after calculations) at the $\overline{CTG}$ input with the time period required for counter time-out. A negative transition of the $\overline{CTG}$ input enables the counter and starts a counter initialization cycle—provided that other conditions as noted in *Table 7* are satisfied. The counter decrements on each clock signal recognized during or after counter initialization until an interrupt is generated, a write-timer-latches command is issued, or a timer reset condition occurs. It can be seen from *Table 7* that an interrupt condition will be generated if TCR_5 = L and the period of the pulse (single pulse or measured separately repetitive pulses) at the $\overline{CTG}$ input is less than the counter time-out period. If TCR_5 = H, an interrupt is generated if the pulse period is greater than the time-out period.

Assume now with TCR_5 = H that a counter initialization has occurred and that the $\overline{CTG}$ input has returned LOW prior to counter time-out. Since there is no individual interrupt flag generated, this automatically starts a new counter initialization cycle. The process will continue with frequency comparison being performed on each CTG input cycle until the mode is changed, or a cycle is determined to be above the predetermined limit.

Pulse Width Comparison Mode (TCR_3 = H, TCR_4 = H)
This mode is similar to the frequency comparison mode except for the limiting factor being a positive, rather than a negative, transition of the $\overline{CTG}$ input. With TCR_5 = L, an individual interrupt flag will be generated if the L level pulse applied to the $\overline{CTG}$ input is less than the time period required for counter time-out. With TCR_5 = H, the interrupt is generated when the reverse condition is true.

As can be seen in *Table 8,* a positive transition of the $\overline{CTG}$ input disables the counter. With TCR_5 = L, it is therefore possible to directly obtain the width of any pulse causing an interrupt.

Table 7 Frequency Comparison Mode

TCR_3 = H, TCR_4 = L

Control Reg Bit 5 (TCR_5)	Counter Initialization	Counter Enable Flip-Flop Set (CE)	Counter Enable Flip-Flop Reset (CE)	Interrupt Flag Set (I)
L	$\overline{G}\downarrow \cdot \overline{I} \cdot (\overline{CE} + TO \cdot \overline{CE}) + R$	$\overline{G}\downarrow \overline{W} \cdot \overline{R} \cdot \overline{I}$	$W + R + I$	$\overline{G}\downarrow$ before TO
H	$\overline{G}\downarrow \cdot \overline{I} + R$	$\overline{G}\downarrow \overline{W} \cdot \overline{R} \cdot \overline{I}$	$W + R + I$	TO before $\overline{G}\downarrow$

$\overline{I}$ represents the interrupt for a given timer.

Table 8 Pulse Width Comparison Mode

TCR_3 = H, TCR_4 = H

Control Reg Bit 5 (TCR_5)	Counter Initialization	Counter Enable Flip-Flop Set (CE)	Counter Enable Flip-Flop Reset (CE)	Interrupt Flag Set (I)
L	$\overline{G}\downarrow \cdot \overline{I} + R$	$\overline{G}\downarrow \cdot \overline{W} \cdot \overline{R} \cdot \overline{I}$	$W + R + I + G$	$\overline{G}\downarrow$ before TO
H	$\overline{G}\downarrow \cdot \overline{I} + R$	$\overline{G}\downarrow\uparrow \overline{W} \cdot \overline{R} \cdot \overline{I}$	$W + R + I + G$	TO before $\overline{G}\uparrow$

Differences Between the F6840 and the F6846 Timers

1. Control Registers 1 and 3 are buried (access through Control Register 2 only) in the F6840 timer. In the F6846 all registers are directly accessible.

2. The F6840 has a dual 8-bit continuous mode for generating non-symmetrical waveforms. The F6846 has a cascaded one-shot mode which can accomplish the same function, but also allows the user to generate waveforms longer than one time-out.

3. Because of the different modes, there is a difference in the control registers between the F6840 and the F6846. (See *Table 9*).

Composite Status Register

The Composite Status Register (CSR) is a read-only register which is shared by the timer and the peripheral data port of the F6846. Three individual interrupt flags in the register are set directly via the appropriate conditions in the timer or peripheral port. The composite interrupt flag, and the $\overline{IRQ}$ output, respond to these individual interrupts only if corresponding enable bits are set in the appropriate control registers. The sequence of assertion is not detected. Setting TCR_6 while CSR_0 is HIGH will cause CSR_7 to be set, for example.

The composite interrupt flag (CSR_7) is clear only if all enabled individual interrupt flags are clear. The conditions for clearing CSR_1 and CSR_2 are detailed in a later section. The timer interrupt flag (CSR_0) is cleared under the following conditions:

1. Timer Reset—Internal Reset bit (TCR_0 = H) or External Reset = L

2. Any Counter Initialization condition

3. A write-timer-latches command if time interval modes (TCR_3 = H) are being used.

4. A read-timer-counter command, provided this is preceded by a read Composite Status Register while CSR_0 is set. This latter condition prevents missing an interrupt request generated after reading the status register and prior to reading the counter.

The remaining bits of the Composite Status Register (CSR_3-CSR_6) are unused. They default to a logic L when read.

5

Composite Status Register and Associated Logic

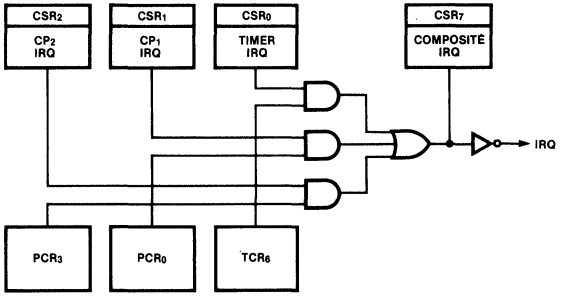
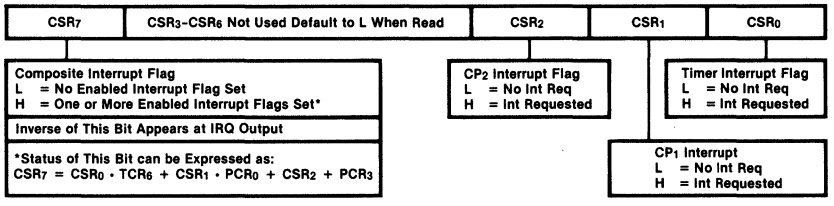


Table 9 F6840 and F6846 Control Register Comparison

Control Register Bit	F6840	F6846
2	16-bit or dual 8-bit mode control	÷8 prescale enable
7	Output enable (all modes)	Output next state (cascaded one-shot mode only), output enable all other modes
0	R1 internal reset R2 control register select R3 timer 3 clock control	Internal reset

Table 10 Composite Status Register Format



I/O Operation

Parallel Peripheral Port

The peripheral port of the F6846 contains eight Peripheral Data lines (P_0-P_7), two peripheral control lines (CP_1 and CP_2), a Peripheral Data Direction Register, a Peripheral Data Register, and a Peripheral Control Register. The port also directly affects two bits (CSR_1 and CSR_2) of the Composite Status Register.

The peripheral port is similar to the "B" side of a PIA (F6821) with the following exceptions:

1. All registers are directly accessible in the F6846. Data direction and peripheral data in the F6821 are located at the same address, with bit 2 of the control register used for register selection.

2. Peripheral Control Register bit 2 (PCR_2) of the F6846 is used to select an optional input latch function. This option is not available with F6821 PIAs.

3. Interrupt flags are located in the F6846 Composite Status Register rather than bits 6 and 7 of the control register as used in the F6821.

4. Interrupt flags are cleared in the F6821 by reading data from the Peripheral Data Register. F6846 interrupt flags are cleared by either reading or writing to the Peripheral Data Register provided that this sequence is followed: a. flag set, b. read Composite Status Register, c. read/write Peripheral Data Register.

5. Bit 6 of the F6846 Peripheral Control Register is not used. Bit 7 (PCR_7) is an internal reset bit not available on the F6821.

6. The Peripheral Data lines (and CP_2) of the F6846 feature internal current limiting which allows them to directly drive the base of Darlington npn transistors.

Peripheral Data Direction Register

The MPU can write directly to this 8-bit register to configure the Peripheral Data lines as either inputs or outputs. A particular bit within the register (DDR_n) is used to control the corresponding Peripheral Data line (P_n). With DDR_n = L, P_n becomes an input; if DDR_n = H, P_n is an output. For example, writing Hex $OF into the Peripheral Data Direction Register results in P_0 through P_3 becoming outputs and P_4 through P_7 inputs. Hex $55 in the Peripheral Data Direction Register results in alternate output and inputs at the parallel port.

Peripheral Data Register

This 8-bit register is used for transferring data between the peripheral data port and the MPU. Any bit corresponding to an output line will be used to drive the output buffer associated with that line. Data in these output bits are normally provided by an MPU write function. (Input bits, those associated with input lines, are unchanged by a write command.) Any input bit will reflect the state of the associated input line if the input latch function is deselected. If the Peripheral Control Register is programmed to provide input latching, the input bit will retain the state at the time CP_1 was activated until the Peripheral Data Register is read by the MPU.

Peripheral Control Register

This 8-bit register is used to control the reset function as well as for selection of optional functions of the two peripheral control lines (CP_1 and CP_2). The Peripheral Control Register functions are outlined in *Table 11.*

Peripheral Port Reset (PCR_7)

Bit 7 of the Peripheral Control Register (PCR_7) may be used to initialize the peripheral section of the F6846. When this bit is set HIGH, the Peripheral Data Register, the Peripheral Data Direction Register, and the interrupt flags associated with the peripheral port (CSR_1 and CSR_2) are all cleared. Other bits in the Peripheral Control Register are not affected by PCR_7.

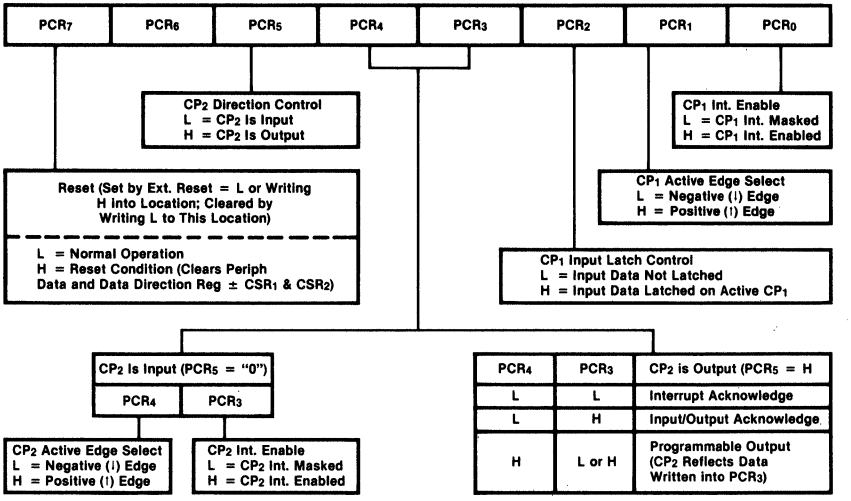
PCR_7 is set by either a logic L at the external $\overline{RESET}$ input or under program control by writing an H into the location. In any case, PCR_7 may be cleared only by writing an L into the location while $\overline{RESET}$ is HIGH. The bit must be cleared to activate the port.

Control of Peripheral Interrupt Line (CP_1)

CP_1 may be used as an interrupt request to the F6846, as a strobe to allow latching of input data, or both. In any case, the input can be programmed to be activated by either a positive or negative transition of the signal. These options are selected via Peripheral Control Register bits PCR_0, PCR_1 and PCR_2.

Peripheral Control Register bit 0 (PCR_0) is used to enable the interrupt transfer circuitry of the F6846. Regardless of the state of PCR_0, an active transition of CP_1 causes the Composite Status Register bit 1 (CSR_1) to be set. If PCR_0 = H, this interrupt will be reflected in the composite interrupt flag (CSR_7), and thus at the $\overline{IRQ}$ output. CSR_1 is cleared by a peripheral port reset condition or by either reading or writing to the Peripheral Data Register after the Composite Status Register is

Table 11 Peripheral Control Register Format (Expanded)



read. The latter alternative is conditional; CSR_1 must have a logic H when the Composite Status Register was last read. This preculudes inadvertent clearing of interrupt flags generated between the time the Composite Status Register is read and the manipulation of peripheral data.

Peripheral Control Register bit 1 (PCR_1) is used to select the edge which activates CP_1. When PCR_1 = H, CP_1 is active on negative transitions (HIGH-to-LOW). LOW-to-HIGH transitions are sensed by CP_1 when PCR_1 = H.

In addition to its use as an interrupt input, CP_1 can be used as a strobe to capture input data in an internal latch. This option is selected by writing a HIGH into Peripheral Control Register bit 2 (PCR_2). In operation, the data at the pins designated by the Peripheral Data Direction Register as inputs will be captured by an active transition of CP_1. An MPU read of the Peripheral Data Register will result in the captured data being transferred to the MPU; it also releases the latch to allow capture of new data. Note that successive active transitions with no read-peripheral-data command between does not update the input latch. Also, it should be noted that use of the input latch function (which can be deselected by writing an L into PCR_2) has no effect on output data. It also does not affect interrupt function of CP_1.

Control of Peripheral Control Line (CP_2)

CP_2 may be used as an input by writing an L into PCR_5. In this configuration, CP_2 becomes a dual of CP_1 in regard to generation of interrupts. An active transition (as selected by PCR_4) causes bit 2 of the Composite Status Register to be set. PCR_3 is then used to select whether the CP_2 transition is to cause CSR_7 to be set and, thereby, cause $\overline{IRQ}$ to go LOW. CP_2 has no effect on the input latch function of the F6846.

Writing an H into PCR_5 causes CP_2 to function as an output. PCR_4 then determines whether CP_2 is to be used in a handshake or programmable output mode. With PCR_4 = H, CP_2 will merely reflect the data written into PCR_3. Since this can readily be changed under program control, this mode allows CP_2 to be a programmable output line in much the same manner as those lines selected as outputs by the Peripheral Data Direction Register.

The handshaking mode (PCR_5 = H, PCR_4 = L) allows CP_2 to perform one of two functions as selected by PCR_3. With PCR_3 = H, CP_2 will go LOW on the first Enable (system $\phi 2$) positive transition after a read or write to the Peripheral Data Register. This input/output acknowledge signal is released (returns HIGH) on the next positive transition of the enable signal.

In the interrupt acknowledge mode (PCR_5 = H, PCR_4 = PCR_3 = L), CP_2 is set when CSR_2 is set by an active transition of CP_1. It is released (goes LOW) on the first positive transition of Enable after CSR_1 has been cleared via an MPU read or write to the Peripheral Data Register. (Note that the previously described conditions for clearing CSR_1 still apply.)

Restart Sequence

A typical restart sequence for the F6846 will include initialization of both the Peripheral Control and Peripheral Data Direction Registers of the parallel port. It is necessary to set up the Peripheral Control Register first, since PCR_7 = L is a condition for writing data into the Peripheral Data Direction Register. (A logic L at the external $\overline{RESET}$ input automatically sets PCR_7).

Absolute Maximum Ratings

Supply Voltage, V_{CC}	−0.3 V, +7.0 V
Input Voltage, V_{IN}	−0.3 V, +7.0 V
Operating Temperature, T_A	0 °C, +70 °C
Storage Temperature, T_{stg}	−55 °C, +150 °C
Thermal Resistance, ϕ_{JA}	70 °C/W

Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Electrical Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = 0 to +70°C unless otherwise noted

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
V_{IH}	Input HIGH Voltage, All Inputs	2.0			V	
V_{IL}	Input LOW Voltage, All Inputs	−0.3		0.8	V	
V_{OS}	Clock Input HIGH	V_{CC} −0.5		V_{CC} +0.5	V	
	Clock Input LOW	V_{SS} −0.5		V_{SS} +0.5		
V_{OH}	Output HIGH Voltage D_0-D_7 Other Outputs	 2.4 2.4			V	 I_{Load} = −205 μA I_{Load} = −200 μA
V_{OL}	Output LOW Voltage D_0-D_7 Other Outputs			 0.4 0.4	V	 I_{Load} = 1.6 mA I_{Load} = 3.2 mA
I_{IN}	Input Leakage Current CP_1, $\overline{CTG}$, $\overline{CTC}$, E A_0-A_{10} R/$\overline{W}$, $\overline{RESET}$, CS_0, CS_1		 1.0	 2.5	 μA	 V_{IN} = 0 to 5.25 V
I_{TSI}	3-State (OFF State) Input Current D_0-D_7, P_0-P_7, CP_2		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
I_{LOH}	Output Leakage Current (OFF State) $\overline{IRQ}$			10	μA	V_{OH} = 2.4 V
I_{OL}	Output LOW Current (Sinking) D_0-D_7 Other Outputs	 1.6 3.2			mA	V_{OL} = 0.4 V
I_{OH}	Output HIGH Current (Sourcing) D_0-D_7 Other Outputs	 −205 −200			μA	V_{OH} = 2.4 V
	CP_2, P_0-P_7	1.0		−10	mA	V_O = 1.5 V for driving other than TTL
P_D	Power Dissipation			1000	mW	
C_{IN}	Input Capacitance E D_0-D_7 All Other Inputs			 200 12.5 7.5	 pF	 V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance P_0-P_7, CP_2, CTO $\overline{IRQ}$			 10 5.0	 pF	 V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
f	Operating Frequency	0.1		1.0	MHz	
$t_{CYC}E$	Clock Cycle Time, E	1.0			μs	
t_{RL}	Clock $\overline{RESET}$ LOW Time	2.0			μs	
t_{IR}	Clock Interrupt Release Time			1.6	μs	*Figure 5*

5

Read/Write Timing *(Figures 1, 2)*

Symbol	Characteristic	Min	Typ	Max	Unit
PW_{EL}	Enable Pulse Width, LOW	430			ns
PW_{EH}	Enable Pulse Width, HIGH	430			ns
t_{AS}	Set-up Time (Address CS_0, CS_1, R/$\overline{W}$)	160			ns
t_{DDR}	Data Delay Time			320	ns
t_H	Data Hold Time	10			ns
t_{AH}	Address Hold Time	10			ns
t_{Er}, t_{Ef}	Rise and Fall Time			25	ns
t_{DSW}	Data Set-up Time	195			ns

Bus Timing Peripheral I/O Lines *(Figures 3, 4, 6, 7 and 11)*

Symbol	Characteristic	Min	Typ	Max	Unit
t_{PDSU}	Peripheral Data Set-up	200			ns
t_{Pr}, t_{Pf}	Rise and Fall Times CP_1, CP_2			1.0	μs
t_{CP2}	Delay Time E to CP_2 Fall			1.0	μs
t_{DC}	Delay Time I/O Data CP_2 Fall	20			ns
t_{RS1}	Delay Time E to CP_2 Rise			1.0	μs
t_{RS2}	Delay Time CP_1 to CP_2 Rise			2.0	μs
t_{PDW}	Peripheral Data Delay			1.0	μs
t_{PSU}	Peripheral Data Set-up Time for Latch	100			ns
t_{PDH}	Peripheral Data Hold Time for Latch	15			ns

Timer-Counter Lines *(Figures 8, 9, and 10)*

Symbol	Characteristic	Min	Typ	Max	Unit
t_{Cr}, t_{Cf}	Input Rise and Fall Time, $\overline{CTC}$ and $\overline{CTG}$			100	ns
t_{PWH}	Input Pulse Width, HIGH (Asynchronous Mode)	t_{cyc} + 250			ns
t_{PWL}	Input Pulse Width, LOW (Asynchronous Mode)	t_{cyc} + 250			ns
t_{su}	Input Set-up Time (Synchronous Mode)	200			ns
t_{hd}	Input Hold Time (Synchronous Mode)	50			ns
t_{CTO}	Output Delay			1.0	μs

Fig. 1 Bus Read Timing (Read Information from F6848)

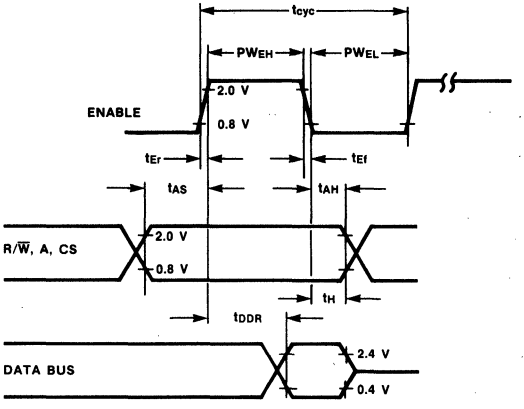


Fig. 2 Bus Write Timing (Write Information from MPU)

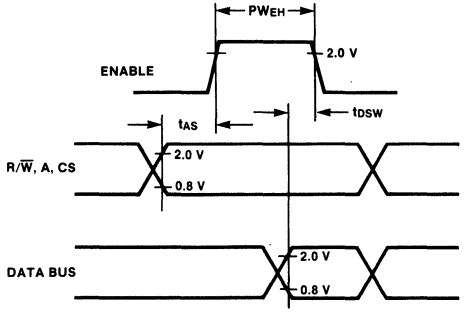


Fig. 3 Peripheral Port Latch Set-up and Hold Time

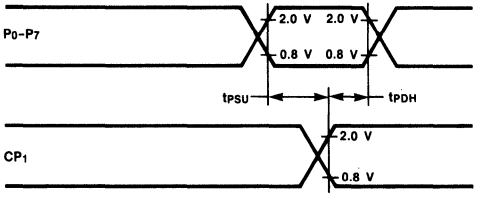
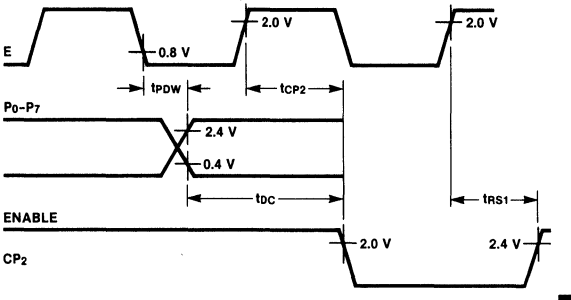


Fig. 4 Peripheral Data and CP_2 Delay (Control Mode PCR_5 = H, PCR_4 = L, PCR_3 = H)



Note
CP_2 goes LOW as the result of positive transition of the second $\phi 2$ pulse

Fig. 5 IRQ Release Time

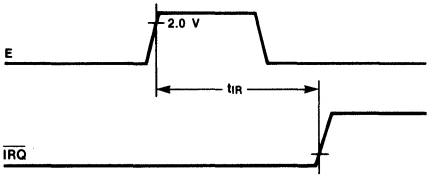


Fig. 6 Peripheral Port Set-up Time

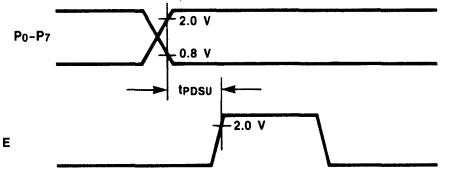
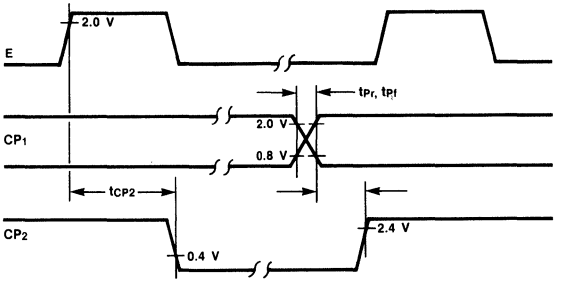


Fig. 7 CP_2 Delay Time (PCR_5 = H, PCR_4 = L, PCR_3 = L)



5

Fig. 8 Input Pulse Widths

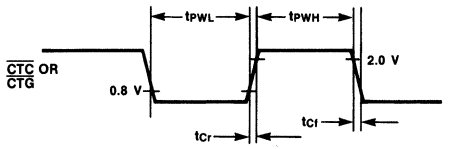
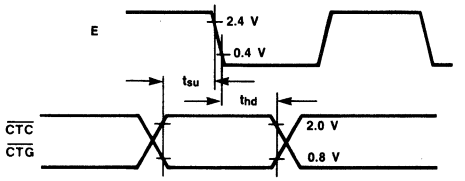


Fig. 9 Input Set-up and Hold Times



Note
This mode is valid only for synchronous operation.

Fig. 10 Output Delay

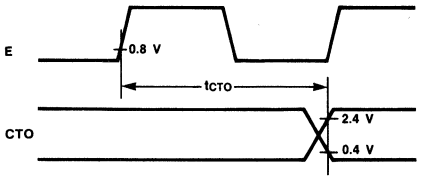
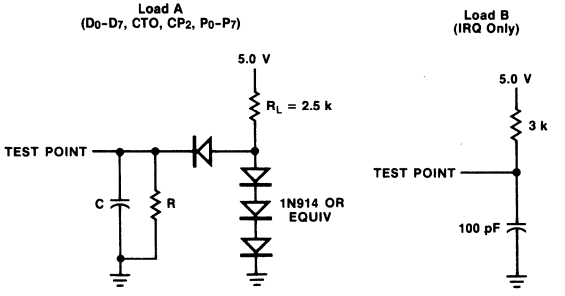


Fig. 11 Bus Timing Test Loads



C = 130 pF for D_0-D_7
= 30 pF for CTO, CP_2, P_0-P_7
R = 11.7 kΩ for D_0-D_7
= 24 kΩ for CTO, CP_2, P_0-P_7

Custom ROM Programming Information
The customer's unique program code pattern may be submitted to Fairchild in several methods. The most convenient and readily verifiable is in the form of 2708, 2716 or 2732 EPROMs. Program code patterns may also be submitted on Fairchild Formulator MKIII floppy disks or on HP cassette tape in Formulator or MIKBUG* format.

Customer Company Name _____
Customer Contact Name _____
Customer Part No. _____
Address _____
Phone No. _____
Fairchild Part No. _____

Fairchild Use Only
SL No. _____
Bid Control No. _____
Field Sales Engineer _____
Date Sent _____

Customer Input Media
- ☐ 2708 EPROM
- ☐ 2716 EPROM
- ☐ 2732 EPROM
- ☐ HP Cassette
  - ☐ Formulator Format

Request for Return Media
- ☐ Listing
- ☐ EPROM (include blank EPROMs)

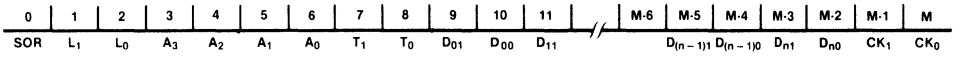
Rom Select*
CS_0 _____
CS_1 _____

I/O Select*
CS_0 _____
CS_1 _____
A_6 _____ (may be unused

Location Select (Select 1 only)
- ☐ A_7 HIGH
- ☐ A_8 HIGH
- ☐ A_9 HIGH
- ☐ A_{10} HIGH
- ☐ Not used

*ROM and I/O Selects must be different.
H = HIGH to Select
L = LOW to Select

Formulator Format



SOR	Start of record defined to be a colon(:)	T_1 T_0	Type field.
L_1 L_0	Length field defined to be the number of packed data bytes per record. Each record is (2 • L) + 11 characters in length inclusive of start of record Length 0 implies end of relocatable module.	D_{01} D_{00}...$D_{(n)1}$ $D_{(n)0}$	Data field.
		CK_1 CK_0	Checksum field defined to be negative modulo 256 summation of all bytes since start of record. A summation of all characters in a record, including the checksum, will result in zero.
A_3 A_2 A_1 A_0	Address Field	All characters other than SOR are ASCII hexadecimal (0-9, A-F).	

5

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6846P, S	0 °C to +70 °C

P = Plastic Package; S = Ceramic Package

FAIRCHILD
A Schlumberger Company

F6847
Video Display Generator

Microprocessor Product

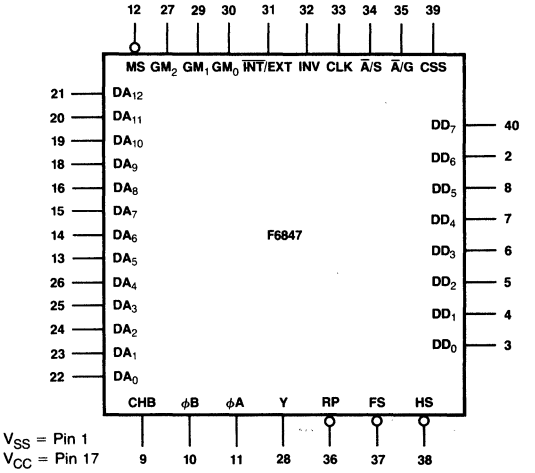
Description

The Fairchild F6847 Video Display Generator (VDG) provides a means of interfacing the Fairchild F6800 microprocessor family (or similar products) to a commercially available color or black-and-white television receiver. Applications of the VDG include video games, bioengineering displays, education, communications, and any instance in which graphics are required.

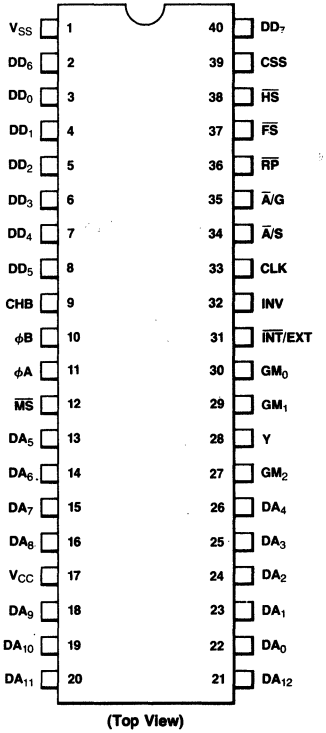
The VDG reads data from memory and produces a composite video signal that allows the generation of alphanumeric or graphic displays. The generated composite video may be up-modulated to either channel 3 or 4 by using a suitable rf modulator. The up-modulated signal is suitable for application to the antenna of a color TV. *Figure 1* illustrates a typical TV game application.

- **Generates Four Different Alphanumeric Display Modes and Eight Graphic Display Modes**
- **Compatible With the F6800 Family**
- **The Alphanumeric Modes Display 32 Characters per Line by 16 Lines.**
- **An Internal Multiplexer Allows the Use of Either the Internal ROM or an External Character Generator.**
- **An External Character Generator Can Be Used to Extend the Internal Character Set for "Limited Graphic" Shapes.**
- **One Display Mode Offers 8-Color 64 x 32 Density Graphics in an Alphanumeric Display Mode.**
- **One Display Mode Offers 4-Color 64 x 48 Density Graphics in an Alphanumeric Display Mode.**
- **All Alphanumeric Modes Have a Selectable Video Inverse.**
- **Generates Full Video Signal**
- **Generates R-Y and B-Y Signals for External Color Modulator**
- **Full Graphic Modes Offer 64 x 64, 128 x 64, 128 x 96, 128 x 192, or 256 x 192 Densities.**
- **Full Graphic Modes Allow 2-Color or 4-Color Data Structures.**
- **Full Graphic Modes Use One of Two 4-Color Sets or One of Two 2-Color Sets.**

Logic Symbol

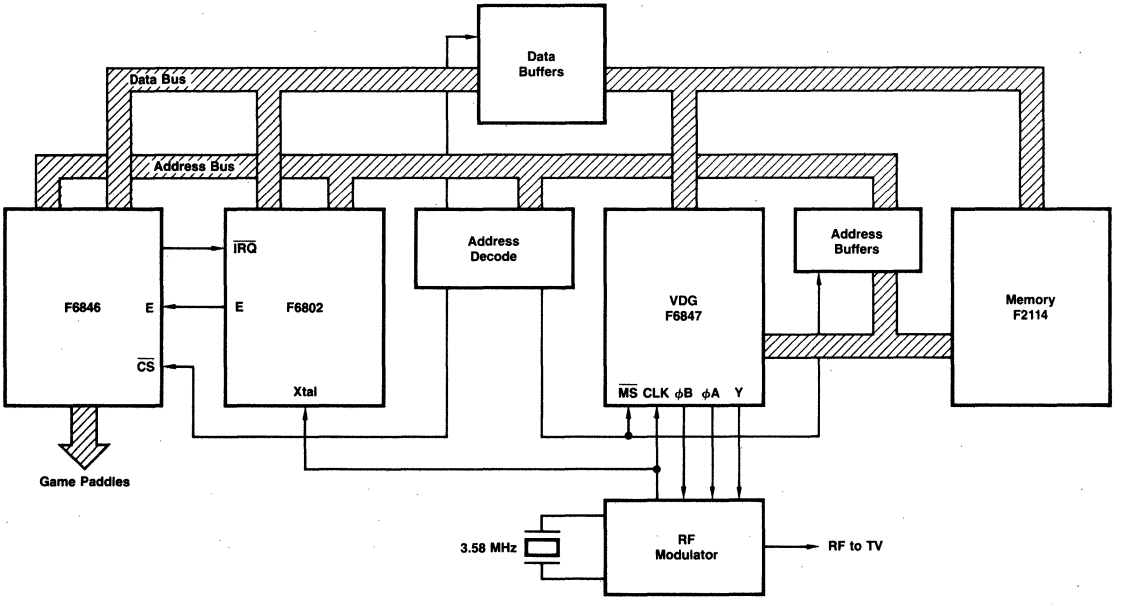


Connection Diagram
40-Pin DIP



(Top View)

Fig. 1. Block Diagram of Use of the VDG in a TV Game



Pin Functions

V_{CC}	+5 V
V_{SS}	Ground
CLK	Color burst clock 3.58 MHz (input)
$DA_0 - DA_{12}$	Address lines to display memory, high impedance during memory select (MS)
$DD_0 - DD_5$	Data from display memory RAM or ROM
DD_6, DD_7	Data from display memory in graphic mode; data also in alpha external mode; color data in alpha semigraphic-4 or -6 mode
φA, φB, Y	Chrominance and luminance analog (R-Y, B-Y, Y) output to rf modulator
CHB	Chroma Bias; reference φA and φB levels
$\overline{RP}$	Row Preset; output to provide timing for external character generator
$\overline{HS}$	Horizontal Sync; output to provide timing for external character generator
INV	Inverts video in all alpha modes
$\overline{INT}$/EXT	Switches to external ROM in alpha mode and between alpha semigraphic-4 and alpha semigraphic-6 in semigraphics mode
$\overline{A}$/S	Alpha/Semigraphics; selects between alpha and semigraphics in alpha mode
$\overline{MS}$	Memory Select; forces VDG address buffers to high-impedance state
$\overline{A}$/G	Switches between alpha and graphic modes
$\overline{FS}$	Field Synchronization; goes low at bottom of active display area
CSS	Color Set Select; selects between two alpha display colors or between two color sets in semigraphics-6 and full graphics mode
$GM_0 - GM_2$	Graphic Mode Select; selects one of eight graphic modes

VDG Signal Descriptions

Address Outputs (DA_0 – DA_{12})
Thirteen address lines are used by the VDG to scan the display memory. The starting address of the display memory is located at the upper left corner of the display screen. As the television sweeps from left to right and top to bottom, the VDG increments the RAM display address. These lines are TTL-compatible and may be forced into a high-impedance state whenever the $\overline{MS}$ pin goes LOW.

Data Inputs (DD_0 – DD_7)
Eight TTL-compatible data lines are used to input data from the RAM to be processed by the VDG. The data is interpreted and transformed into luminance Y (pin 28) and color outputs ϕA and ϕB (pin 11 and pin 10).

Power Inputs
V_{CC} requires +5 volts. V_{SS} requires zero volts and is normally ground. (The tolerance and current requirements of the VDG are specified in the *DC Characteristics* table.)

Video Outputs (ϕA, ϕB, Y, CHB)
These four analog outputs are used to transfer luminance and color information to a standard NTSC color television receiver, either via the rf modulator or directly into Y, ϕA, and ϕB television video inputs.

Luminance (Y) – This six-level analog output contains composite sync, blanking, and four levels of video luminance.

ϕA – This three-level analog output is used in combination with the ϕB and Y outputs to specify one of eight colors.

ϕB – This four-level analog output is used in combination with the ϕA and Y outputs to specify one of eight colors. Additionally, one analog level is used to specify the time of the color burst reference signal.

Chroma Bias (CHB) – This pin is an analog output and provides a dc reference corresponding to the quiescent value of ϕA and ϕB. CHB is used to guarantee good thermal tracking and minimize the variation between the parts.

Synchronizing Inputs ($\overline{MS}$, CLK)

Three-State Control ($\overline{MS}$) – This is a TTL-compatible input that, when LOW, forces the VDG address lines into a high-impedance state. This may be done to allow other devices (such as an MPU) to address the display memory RAM.

Clock (CLK) – The VDG clock input (CLK) requires a 3.579545 MHz (standard) TV crystal frequency square wave. The duty cycle of this clock must be between 45 and 55 percent since it controls the width of alternate dots on the television screen.

Synchronizing Outputs ($\overline{FS}$, $\overline{HS}$, $\overline{RP}$)
Three TTL-compatible outputs provide circuits exterior to the VDG with timing references to the following internal VDG states:

Field Sync ($\overline{FS}$) – The HIGH-to-LOW transition of the $\overline{FS}$ output coincides with the end of active display area. During this time interval, an MPU may have total access to the display RAM without causing undesired flicker on the screen. The LOW-to-HIGH transition of $\overline{FS}$ coincides with the trailing edge of the vertical synchronization pulse.

Horizontal Sync ($\overline{HS}$) – The HIGH-to-LOW transition of the $\overline{HS}$ output coincides with the leading edge of the horizontal snyc pulse portion of the VDG luminance (Y) output.

Row Preset ($\overline{RP}$) – If desired, an external character generator ROM may be used with the VDG. In this configuration, an external 4-bit counter, used to supply row selection, is clocked by $\overline{HS}$ and cleared by the $\overline{RP}$ signal.

Mode Control Inputs ($\overline{A}$/G, $\overline{A}$/S, $\overline{INT}$/EXT, GM_0, GM_1, GM_2, CSS, INV)
Eight TTL-compatible inputs are used to control the operating mode of the VDG. $\overline{A}$/S, $\overline{INT}$/EXT, CSS and INV may be changed on a character-by-character basis. The CSS pin is used to select between two possible alphanumeric colors when the VDG is in the alphanumeric mode and between two color sets when the VDG is in the semigraphics-6 and full graphic mode. *Table 1* illustrates the various modes that can be obtained using the mode contol lines.

Display Modes

The VDG is capable of generating 12 distinct display modes. The color set selection (CSS) and invert (INV) pins allow variations on certain modes. The VDG displays two alphanumeric modes with two compatible semigraphic modes or one of eight full graphic modes. A detailed description of the various modes of operation follows. A summary of major modes can be found in *Table 2*, and a detailed description of VDG modes can be found in *Table 3*.

Alphanumeric Display Modes
All alphanumeric modes occupy an 8 x 12 dot character matrix box; there are 32 x 16 character boxes per TV frame. Each horizontal dot (dot-clock) corresponds to one-half the period duration of the 3.58 MHz clock, and each vertical dot is one scan line. One of two colors for the lighted dots may be selected by the color set select pin.

Internal Alphanumeric Mode – In the internal alphanumeric mode, an internal ROM will generate 64 ASCII display characters in a standard 5 x 7 box. Six bits of the 8-bit data word are used for the ASCII character generator; the two bits not used can be used to implement inverse video or color switching on a character-by-character basis. A 512-word display memory is required for this class of display.

External Alphanumeric Mode – In the external alphanumeric mode, an external character generator may be used to generate custom character sets of up to 256 separate 8 x 12 dot characters, each defined by an 8-bit data word. If fewer than eight bits are used for character definition, the remaining bits may be used for inverse video selection or color switching on a character-by-character basis. This display mode also requires a 512-word display memory.

Alpha Semigraphic-4 Mode – The alpha semigraphic-4 mode translates bits 0 through 3 into a 4 x 6 dot element in the standard 8 x 12 dot box. Three data bits may be used to select one of eight colors for the entire character box. The extra bit is available to implement mode switching on-the-fly. A 512-word display memory is required. A density of 64 x 32 elements is available in the display area. The element area is four dot-clocks wide by six lines high.

Alpha Semigraphic-6 Mode – The alpha semigraphic-6 mode maps six 4 x 4 dot elements into the standard 8 x 12 dot alphanumeric box, providing a screen density of 64 x 48 elements. Six bits are used to generate this map and two data bits may be used to select one of four colors in the display box. The element area is four dot-clocks wide by four lines high.

Full Graphic Mode

There are eight full graphic modes available from the VDG. These modes require 1K to 6K bytes of memory. The eight full graphic modes include an outside color border in one of two colors, depending upon the color set select (CSS) pin. The CSS pin selects one of two sets of four colors in the four color graphic modes.

The 64 x 64 Color Graphics Mode (Graphics One C) – The 64 x 64 color graphics mode generates a display matrix 64 elements wide by 64 elements high. Each element may be one of four colors. A 1K x 8 display memory is required. Each pictel equals four dot-clocks by three scan lines.

The 128 x 64 Graphics Mode (Graphics One R) – The 128 x 64 graphics mode generates a matrix 128 elements wide by 64 elements high. Each element may be either On or Off. However, the entire display may be one of two colors, selected by using the color set select pin. A 1K x 8 display memory is required. Each pictel equals two dot-clocks by three scan lines.

The 128 x 64 Color Graphics Mode (Graphics Two C) – The 128 x 64 color graphics mode generates a display matrix 128 elements wide by 64 elements high. Each element may be one of four colors. A 2K x 8 display memory is required. Each pictel equals two dot-clocks by three scan lines.

The 128 x 96 Graphics Mode (Graphics Two R) – The 128 x 96 graphics mode generates a display matrix 128 elements wide by 96 elements high. Each element may be either On or Off. However, the entire display may be one of two colors, selected by using the color set select pin. A 2K x 8 display memory is required. Each pictel equals two dot-clocks by two scan lines.

The 128 x 96 Color Graphics Mode (Graphics Three C) – The 128 x 96 color graphics mode generates a display 128 elements wide by 96 elements high. Each element may be one of four colors. A 3K x 8 display memory is required. Each pictel equals two dot-clocks by two scan lines.

The 128 x 192 Graphics Mode (Graphics Three R) – The 128 x 192 graphics mode generates a display matrix 128 elements wide by 192 elements high. Each element may be either On or Off, but the On elements may be one of two colors, selected with the color set select pin. A 3K x 8 display memory is required. Each pictel equals two dot-clocks by one scan line.

128 x 192 Color Graphics Mode (Graphics Six C) – The 128 x 192 color graphics mode generates a display 128 elements wide by 192 element high. Each element may be one of four colors. A 6K x 8 display memory is required. Each pictel equals two dot-clocks by one scan line.

The 256 x 192 Graphics Mode (Graphics Six R) – The 256 x 192 graphics mode generates a display 256 elements wide by 192 elements high. Each element may be either On or Off, but the On elements may be one of two colors, selected with the color set select pin. A 6K x 8 display memory is required. Each pictel equals one dot-clock by one scan line.

Table 1 Mode Control Inputs

$\overline{A}/G$	$\overline{A}/S$	$\overline{INT}/EXT$	INV	GM_2	GM_1	GM_0	Alpha/Graphic Mode Selected
0	0	0	0	X	X	X	Internal Alphanumeric
0	0	0	1	X	X	X	Internal Alphanumeric Inverted
0	0	1	0	X	X	X	External Alphanumeric
0	0	1	1	X	X	X	External Alphanumeric Inverted
0	1	0	X	X	X	X	Alpha Semigraphic-4
0	1	1	X	X	X	X	Alpha Semigraphic-6
1	X	X	X	0	0	0	64 x 64 Color Graphic
1	X	X	X	0	0	1	128 x 64 Graphic
1	X	X	X	0	1	0	128 x 64 Color Graphic
1	X	X	X	0	1	1	128 x 96 Graphic
1	X	X	X	1	0	0	128 x 96 Color Graphic
1	X	X	X	1	0	1	128 x 192 Graphic
1	X	X	X	1	1	0	128 x 192 Color Graphic
1	X	X	X	1	1	1	256 x 192 Graphic

Table 2 Summary of Major Modes

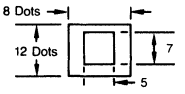
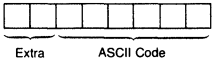
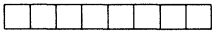
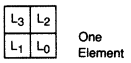
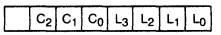
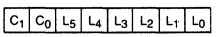
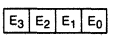
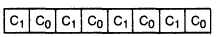
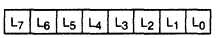
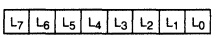
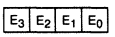
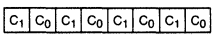
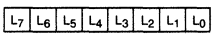
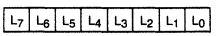
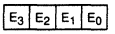
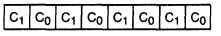
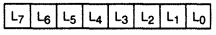
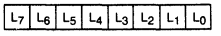
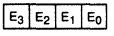
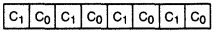
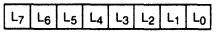
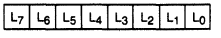
Title	Memory	Colors	Display Elements
Alphanumeric (Internal)	512 x 8	2	
Alphanumeric (External)	512 x 8	2	
Alpha Semigraphic-4	512 x 8	8	Box — Element
Alpha Semigraphic-6	512 x 8	4	Box — Element
64 x 64 Color Graphic	1K x 8	4	Matrix 64 x 64 Elements
128 x 64 Graphic*	1K x 8	2	Matrix 128 Elements Wide by 64 Elements High
128 x 64 Color Graphic	2K x 8	4	
128 x 96 Graphic*	1.5K x 8	2	Matrix 128 Elements Wide by 96 Elements High
128 x 96 Color Graphic	3K x 8	4	
128 x 192 Graphic*	3K x 8	2	Matrix 128 Elements Wide by 192 Elements High
128 x 192 Color Graphic	6K x 8	4	
256 x 192 Graphic*	6K x 8	2	Matrix 256 Elements Wide by 192 Elements High

*Graphic mode turns each element on or off. The color may be one of two.

Table 3 Detailed Description of VDG Modes

VDG Pins									Color		
$\overline{MS}$	$\overline{A/G}$	$\overline{A/S}$	$\overline{INT}$/EXT	GM_2	GM_1	GM_0	CSS	INV	Character Color	Background	Border
1	0	0	0	X	X	X	0	0	Green Black	Black Green	Black
								1			
							1	0	Orange Black	Black Orange	Black
								1			
1	0	0	1	X	X	X	0	0	Green Black	Black Green	Black
								1			
							1	0	Orange Black	Black Orange	Black
								1			
1	0	1	0	X	X	X	X	X	L_x C_2 C_1 C_0 Color 0 X X X Black 1 0 0 0 Green 1 0 0 1 Yellow 1 0 1 0 Blue 1 0 1 1 Red 1 1 0 0 Buff 1 1 0 1 Cyan 1 1 1 0 Magenta 1 1 1 1 Orange		Black
1	0	1	1	X	X	X	0	X	L_x C_1 C_0 Color 0 X X Black 1 0 0 Green 1 0 1 Yellow 1 1 0 Blue 1 1 1 Red 0 X X Black 1 0 0 Buff 1 0 1 Cyan 1 1 0 Magenta 1 1 1 Orange		Black
							1				
1	1	X	X	0	0	0	0	X	C_1 C_0 Color 0 0 Green 0 1 Yellow 1 0 Blue 1 1 Red 0 0 Buff 0 1 Cyan 1 0 Magenta 1 1 Orange		Green
							1				Buff
1	1	X	X	0	0	1	0	X	L_x 0 1	Color Black Green	Green
							1		0 1	Black Buff	Buff
1	1	X	X	0	1	0	0	X	Same Color as Graphics One C		Green
							1				Buff
1	1	X	X	0	1	1	0	X	Same Color as Graphics One R		Green
							1				Buff
1	1	X	X	1	0	0	0	X	Same Color as Graphics One C		Green
							1				Buff
1	1	X	X	1	0	1	0	X	Same Color as Graphics One R		Green
							1				Buff
1	1	X	X	1	1	0	0	X	Same Color as Graphics One C		Green
							1				Buff
1	1	X	X	1	1	1	0	X	Same Color as Graphics One R		Green
							1				Buff

Table 3 Detailed Description of VDG Modes (Cont.)

TV Screen		VDG Data Bus	Comments
Display Mode	**Detail**		
32 Characters in Columns 16 Characters in Rows			The internal alphanumeric mode uses an internal character generator that contains the following five dot by seven dot characters: @ ABCDEF GHIJKLMNOPQRSTUVWXYZ [ ] ↑ ← SP !"#$%&'()*+,-./0123456789:;<=>?. The 6-bit ASCII code leaves two bits free; these may be externally connected to the mode pins ($\overline{A}$/G, $\overline{A}$/S, $\overline{INT}$/EXT, GM_2, GM_1, GM_0, CSS or INV).
32 Characters in Columns 16 Characters in Rows			The external alphanumeric mode uses an external character generator as well as a row counter. Thus, custom character fonts are graphic symbol sets with up to 256 different 8 dot X 12 dot "characters" that may be displayed.
64 Display Elements in Columns 32 Display Elements in Rows			The semigraphic-4 mode uses an internal "coarse graphics" generator in which a rectangle (8 dots by 12 dots) is divided into four equal parts. The luminance of each part is determined by a corresponding bit on the VDG data bus. The color of illuminated parts is determined by three bits.
64 Display Elements in Columns 48 Display Elements in Rows			The semigraphic-6 mode is similar to the semigraphic-4 mode with the following differences. The 8 dot by 12 dot rectangle is divided into six equal parts. Color is determined by the two remaining bits.
64 Display Elements in Columns 64 Display Elements in Rows			The graphics one C mode uses a maximum of 1024 bytes of display RAM in which one pair of bits specifies one picture element.
128 Display Elements in Columns 64 Display Elements in Rows			The graphics one R mode uses a maximum of 1024 bytes of display RAM in which one bit specifies one picture element.
128 Display Elements in Columns 64 Display Elements in Rows			The graphics two C mode uses a maximum of 2048 bytes of display RAM in which one pair of bit specifies one picture element.
128 Display Elements in Columns 96 Display Elements in Rows			The graphics two R mode uses a maximum of 1536 bytes of display RAM in which one bit specifies one picture element.
128 Display Elements in Columns 96 Display Elements in Rows			The graphics three C mode uses a maximum of 3072 bytes of display RAM in which one pair of bytes specifies one picture element.
128 Display Elements in Columns 192 Display Elements in Rows			The graphics three R mode uses a maximum of 3072 bytes of display RAM in which one bit specifies on picture element.
128 Display Elements in Columns 192 Display Elements in Rows			The graphics six C mode uses a maximum of 6144 bytes of display RAM in which one pair of bit specifies one picture element.
256 Display Elements in Columns 192 Display Elements in Rows			The graphics six R mode uses a maximum of 6144 bytes of display RAM in which one bit specifies one picture element.

5

Absolute Maximum Ratings

Supply Voltage, V_{CC}	−0.3 V, +7.0 V
Input Voltage, any Pin, V_{IN}	−0.3 V, +7.0 V
Operating Temperature Range, T_A	0°C, +70°C
Storage Temperature Range, T_{STG}	−65°C, +150°C
Power Dissipation, P_D	945 mW

Stresses greater than those listed may cause permanent damage to the device. This is a stress rating only, and functional operation of the device under these or any other conditions above those indicated in the operational sections of this data sheet is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0.0 V, T_A = 0°C to +70°C, unless otherwise noted

Symbol	Characteristic	Min	Typ	Max	Unit	Conditions
V_{IH}	Input HIGH Voltage CLK Other Inputs	 V_{SS} + 2.4 V_{SS} + 2.0		 V_{CC} V_{CC}	Vdc	
V_{IL}	Input LOW Voltage CLK Other Inputs	 V_{SS} − 0.3 V_{SS} − 0.3		 V_{SS} + 0.4 V_{SS} + 0.8	Vdc	
I_{in}	Input Leakage Current CLK, GM_0-GM_2, INV, $\overline{INT}$/EXT, $\overline{MS}$, V_{SS}, DD_0-DD_7, $\overline{A}$/S, $\overline{A}$/G			2.5	μAdc	
I_{LO}	Three-State (OFF State) Input Current DA_0-DA_{12}			10	μAdc	
V_{OH}	Output HIGH Voltage $\overline{RP}$, $\overline{HS}$, $\overline{FS}$	2.4			Vdc	C_{Load} = 30 pF I_{Load} = −100 μA
V_{OH}	Output HIGH Voltage DA_0-DA_{12}	2.4			Vdc	C_{Load} = 55 pF I_{Load} = −100 μA
V_{OL}	Output LOW Voltage $\overline{RP}$, $\overline{HS}$, $\overline{FS}$,			V_{SS} + 0.4	Vdc	C_{Load} = 30 pF I_{Load} = 1.6 mA
V_{OL}	Output LOW Voltage DA_0-DA_{12}			V_{SS} + 0.4	Vdc	C_{Load} = 55 pF I_{Load} = 1.6 mA
I_{OH}	Output HIGH Current (Sourcing) All Outputs (Except ϕA, ϕB, Y, and CHB)	−100			μAdc	V_{OH} = 2.4V
I_{OL}	Output LOW Current (Sinking) All Outputs (Except ϕA, ϕB, Y, and CHB	1.6			mAdc	V_{OL} = 0.4 Vdc
C_{IN}	Input Capacitance All Inputs			7.5	pF	V_{IN} = 0 T_A = 25°C f = 1.0 MHz
V_R	Chroma Bias Voltage		0.3 V_{CC}		Vdc	C_{Load} = 20 pF R_{Load} = 200 kΩ V_{CC} = 4.75 − 5.25 V
I_{CC}	Supply Current		90	114	mAdc	

DC Characteristics (Cont.)

Symbol	Characteristic	Min	Typ	Max	Unit	Conditions
$V_{C\phi A}$	Chroma ϕA Voltage V_{HI} V_O V_{LO}		 $V_R + 0.1\ V_{CC}$ V_R $V_R - 0.1\ V_{CC}$		 Vdc	 C_{Load} = 20 pF R_{Load} = 200 kΩ See *Figure 2*
$V_{C\phi B}$	Chroma ϕB Voltage V_O V_{Burst} V_{LO}		$V_R + 0.1\ V_{CC}$ V_R $V_R - 0.05\ V_{CC}$ $V_R - 0.1\ V_{CC}$		Vdc	C_{Load} = 20 pF R_{Load} = 200 kΩ See *Figure 2*
V_Y	Luminance Y Voltage V_S V_{BLANK} V_{BLACK}		 $0.2\ V_{CC}$ $0.75\ V_S$ $0.7\ V_S$		 Vdc	 C_{Load} = 20 pF R_{Load} = 200 kΩ See *Figure 2*
V_{WL} V_{WM} V_{WH}	Voltage White Low Voltage White Medium Voltage White High		0.62 $0.5\ V_S$ $0.38\ V_S$		Vdc	See *Figure 2*

AC Characteristics V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C

Symbol	Characteristic	Min	Typ	Max	Unit	Conditions
f CLK_{dc}	CLK Frequency CLK Duty Cycle	3.579535 45%	3.579545 50%	3.579555 55%	MHz	
 t_{YA} t_{YB}	Chroma Phase Delay (Measured with Respect to Y Output) ϕA ϕB		 200 200		 ns ns	See *Figure 3C*
t_{ry} t_{fy}	Luminance Rise Time Luminance Fall Time		60 50		ns ns	See *Figure 3D*
 $t_rC\phi A$ $t_fC\phi A$ $t_rC\phi B$ $t_fC\phi B$	Chroma Rise and Fall Times ϕA Rise Time ϕA Fall Time ϕB Rise Time ϕB Fall Time		 60 60 60 60		 ns ns ns ns	See *Figure 3D*
t_{WFS}	Field Sync ($\overline{FS}$) Pulse Width		2.03		ms	See *Figure 3A*
t_{WRP} t_{HSRP}	Row Preset ($\overline{RP}$) Pulse Width Delay from $\overline{HS}$		0.98 0.98		µs µs	See *Figure 3B*
t_{WHS}	Horizontal Sync ($\overline{HS}$)		4.9		µs	See *Figure 3B*

5

Fig. 2 Video and Chrominance Relationships Output Waveform

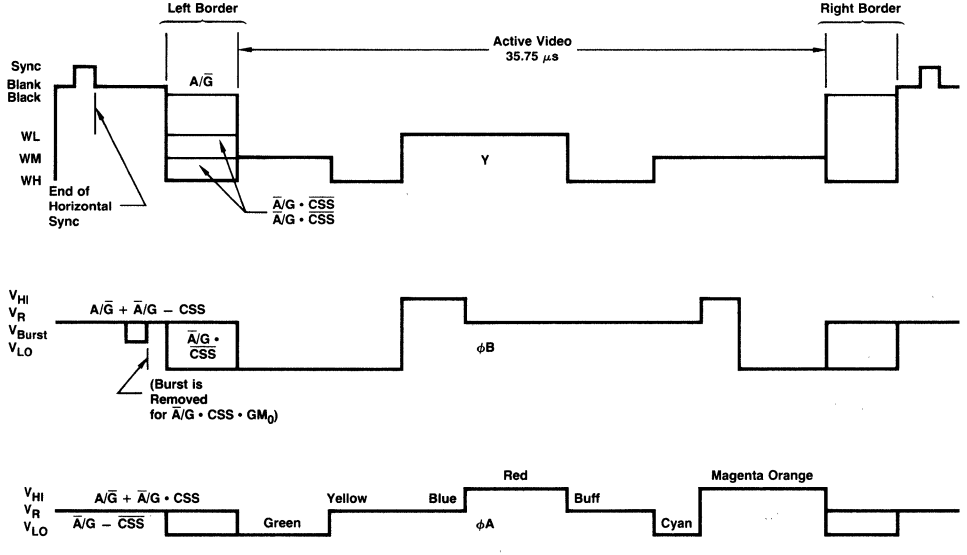
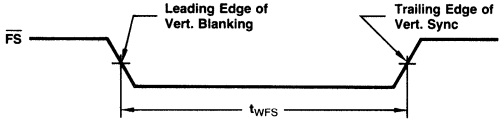
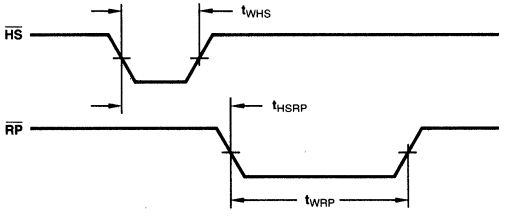


Fig. 3 Timing Diagrams

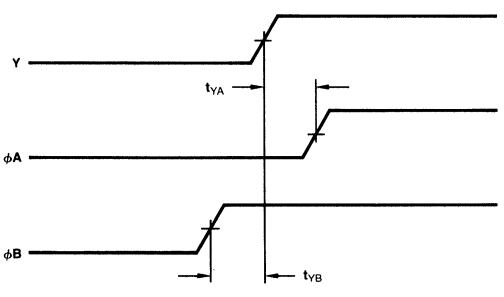
a. Field Sync



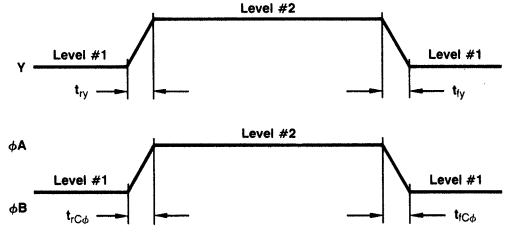
b. Row Preset



c. Chroma Phase Delay



d. Video and Fall Times



Ordering Information

Order Code	Temperature Range
F6847P, S	0°C to +70°C

P = Plastic Package
S = Ceramic Package

FAIRCHILD
A Schlumberger Company

F6850/F68A50/F68B50 Asynchronous Communications Interface Adapter (ACIA)

Microprocessor Product

Description

The F6850 Asynchronous Communications Interface Adapter (ACIA) provides the data formatting and control to interface serial asynchronous data communications information to bus-organized systems, such as the F6800 microprocessing unit (MPU).

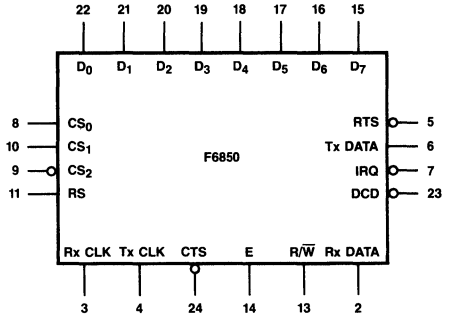
The bus interface of the F6850 includes select, enable read/write, interrupt, and bus interface logic to allow data transfer over an 8-bit bidirectional data bus. The parallel data of the bus system is serially transmitted and received by the asynchronous data interface, with proper formatting and error checking. The functional configuration of the ACIA is programmed via the data bus during system initialization. A programmable control register provides variable word lengths, clock division ratios, transmit control, receive control, and interrupt control. For peripheral or modem operation, three control lines are provided. These lines allow the ACIA to interface directly with a 0-600 bps modem.

- **8- and 9-Bit Transmission**
- **Optional Even and Odd Parity**
- **Parity, Overrun, and Framing Error Checking**
- **Programmable Control Register**
- **Optional ÷1, ÷16, and ÷64 Clock Modes**
- **Up to 1.0 Mbps Transmission**
- **False Start Bit Deletion**
- **Peripheral/Modem Control Functions**
- **Double Buffered**
- **One or Two Stop Bit Operation**

Pin Functions

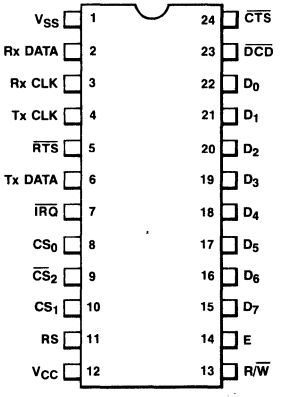
D_0–D_7	Bidirectional Data Lines
Rx DATA	Receive Data Input
Rx CLK	Receive Clock Input
Tx CLK	Transmit Clock Input
CS_0, CS_1, $\overline{CS_2}$	Chip Select Inputs
$\overline{RS}$	Register Select Input
$\overline{CTS}$	Clear-to-Send Input
E	Enable Input
$R/\overline{W}$	Read/Write Input
$\overline{RTS}$	Request-to-Send Output
Tx DATA	Transmit Data Output
$\overline{IRQ}$	Interrupt Request Output
$\overline{DCD}$	Data Carrier Detect Output

Logic Symbol



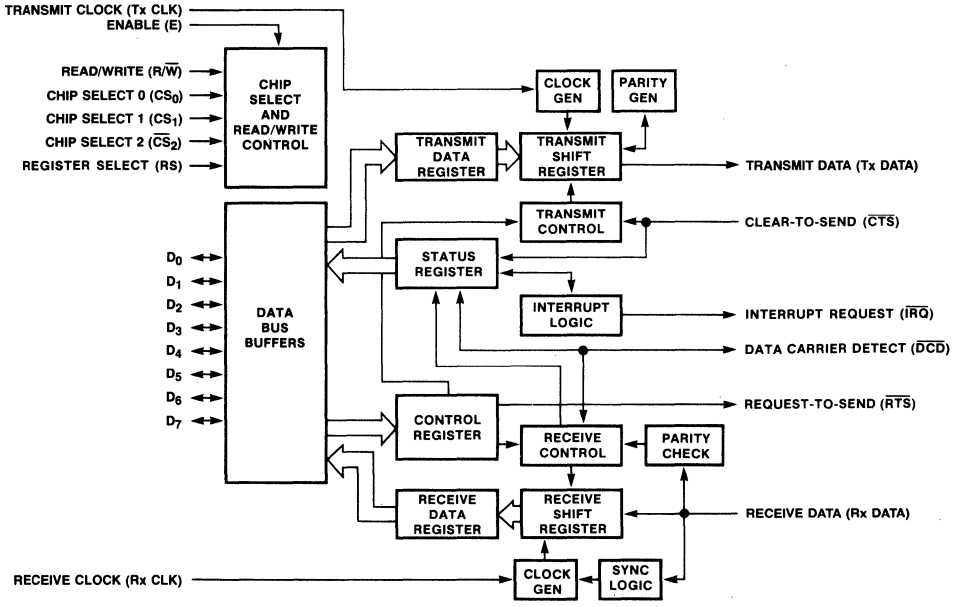
V_{CC} = Pin 12
V_{SS} = Pin 1

Connection Diagram
24-Pin DIP



(Top View)

Block Diagram



Functional Description

At the bus interface, the ACIA appears as two addressable memory locations. Internally, there are four registers: two read-only and two write-only. The read-only registers are status and receive data; the write-only registers are control and transmit data. The serial interface consists of serial input and output lines with independent clocks, and three peripheral/modem control lines.

Power On/Master Reset

The master reset (CR_0, CR_1) should be set during system initialization to ensure the reset condition and prepare for programming the ACIA functional configuration when the communications channel is required. Control bits CR_5 and CR_6 should also be programmed to define the state of the request-to-send ($\overline{RTS}$) output whenever master reset is utilized. The ACIA also contains internal power-on reset logic to detect the power line turn-on transition and hold the chip in a reset state to prevent erroneous output transitions prior to initialization. This circuitry depends on clean power turn-on transitions. The power-on reset is released by means of the bus-programmed master reset, which must be applied prior to operating the ACIA. After master resetting the ACIA, the programmable control register can be set for a number of options, such as variable clock divider ratios, variable word length, one or two stop bits, and parity (even, odd, or none).

Transmit

A typical transmitting sequence consists of reading the ACIA status register either as a result of an interrupt or in turn in a polling sequence. A character may be written into the transmit data register if the status read operation has indicated that the transmit data register is empty. This character is transferred to a shift register, where it is serialized and transmitted from the transmit data (Tx DATA) output preceded by a start bit and followed by one or two stop bits. Internal parity (odd or even) can be optionally added to the character, and occurs between the last data bit and the first stop bit. After the first character is written in the data register, the status register can be read again to check for a transmit data register empty condition and current peripheral status. If the register is empty, another character can be loaded for transmission even though the first character is in the process of being transmitted (because of double buffering). The second character is transferred automatically into the shift register when the first character transmission is completed. This sequence continues until all the characters have been transmitted.

Receive

Data is received from a peripheral by means of the receive data (Rx DATA) input. A divide-by-one clock ratio is provided for an externally synchronized clock (to its data) while the divide-by-16 and -64 ratios are provided for internal synchronization. Bit

synchronization in the divide-by-16 and -64 modes is initiated by the detection of 8 or 32 LOW samples, respectively, on the receive data line. False start bit deletion capability ensures that a full half bit of a start bit has been received before the internal clock is synchronized to the bit time. As a character is being received, parity (odd or even) is checked and the error indication made available in the status register along with framing error, overrun error, and receive data register full. In a typical receiving sequence, the status register is read to determine if a character has been received from a peripheral. If the receive data is full, the character is placed on the 8-bit ACIA bus when a read data command is received from the MPU. When parity has been selected for an 8-bit word (seven bits plus parity), the receiver strips the parity bit ($D_7 = 0$) so that data alone is transferred to the MPU. This feature reduces MPU programming. The status register can be read again to determine when another character is available in the receive data register. The receiver is also double buffered so that a character can be read from the data register as another character is being received in the shift register. The above sequence continues until all characters have been received.

Input/Output Functions
The ACIA interfaces to the F6800 MPU through an 8-bit bidirectional data bus, three chip select lines, a register select line, an interrupt request line, read/write line, and enable line. These signals, in conjunction with F6800 VMA output, permit the MPU to have complete control over the ACIA.

ACIA Bidirectional Data ($D_0 - D_7$)
The bidirectional data lines ($D_0 - D_7$) allow for data transfer between the ACIA and the MPU. The data bus output drivers are 3-state devices that remain in the high-impedance (OFF) state except when the MPU performs an ACIA read operation.

ACIA Enable (E)
The enable signal (E) is a high-impedance, TTL-compatible input that enables the bus input/output data buffers and clocks data to and from the ACIA. This signal normally is a derivative of the F6800 $\phi 2$ clock.

Read/Write (R/$\overline{W}$)
The read/write line is a high-impedance input that is TTL-compatible and is used to control the direction of data flow through the ACIA input/output data bus interface. When R/$\overline{W}$ is HIGH (MPU read cycle), ACIA output drivers are turned on and a selected register is read. When it is LOW, the ACIA output drivers are turned off and the MPU writes into a selected register. Therefore, the R/$\overline{W}$ signal is used to select read-only or write-only registers within the ACIA.

Chip Select (CS_0, CS_1, $\overline{CS_2}$)
These three high-impedance, TTL-compatible input lines are used to address the ACIA. The ACIA is selected when CS_0 and CS_1 are HIGH and $\overline{CS_2}$ is LOW. Transfers of data to and from the ACIA are then performed under the control of the E, R/$\overline{W}$, and RS signals.

Register Select (RS)
The register select line is a high-impedance input that is TTL-compatible. A HIGH level is used to select the transmit/receive data registers and a LOW level the control/status registers. The R/$\overline{W}$ signal line is used in conjunction with RS to select the read-only or write-only register in each register pair.

Interrupt Request ($\overline{IRQ}$)
Interrupt request is a TTL-compatible, open-drain (no internal pull-up), active-LOW output that is used to interrupt the MPU. The $\overline{IRQ}$ output remains LOW as long as the cause of the interrupt is present and the appropriate interrupt enable within the ACIA is set. The IRQ status bit, when HIGH, indicates that the $\overline{IRQ}$ output is in the active state.

Interrupts result from conditions in both the transmitter and receiver sections of the ACIA. The transmitter section causes an interrupt when the transmitter interrupt enabled condition is selected ($CR_5 \cdot \overline{CR_6}$), and the transmit data register empty (TDRE) status bit is HIGH. The TDRE status bit indicates the current status of the transmitter data register except when inhibited by the $\overline{CTS}$ line being HIGH or the ACIA being maintained in the reset condition. The interrupt is cleared by writing data into the transmit data register. The interrupt is masked by disabling the transmitter interrupt via CR_5 or $\overline{CR_6}$, or by the loss of $\overline{CTS}$, which inhibits the TDRE status bit. The receiver section causes an interrupt when the receiver interrupt enable is set and the receive data register full (RDRF) status bit is HIGH, an overrun has occurred, or the data carrier detect ($\overline{DCD}$) line has gone HIGH. An interrupt resulting from the RDRF status bit can be cleared by reading data or resetting the ACIA. Interrupts caused by overrun or loss of $\overline{DCD}$ are cleared by reading the status register after the error condition has occurred and then reading the receive data register or resetting the ACIA. The receiver interrupt is masked by resetting the receiver interrupt enable.

Clock Inputs
Separate high-impedance, TTL-compatible inputs are provided for clocking of transmitted and received data. Clock frequencies of 1, 16, or 64 times the data rate may be selected.

Transmit Clock (Tx CLK)
The transmit clock input is used for the clocking of transmitted data. The transmitter initiates data on the negative transition of the clock.

Receive Clock (Rx CLK)
The receive clock input is used for synchronization of received data. (In the ÷1 mode, the clock and data must be synchronized externally.) The receiver samples the data on the positive transition of the clock.

Serial Input/Output Lines

Receive Data (Rx DATA)
The receive data line is a high-impedance, TTL-compatible input through which data is received in a serial format. Synchronization with a clock for detection of data is accomplished internally when clock rates of 16 or 64 times the bit rate are used.

Transmit Data (Tx DATA)
The transmit data output line transfers serial data to a modem or other peripheral.

Peripheral/Modem Control
The ACIA includes several functions that permit limited control of a peripheral or modem. The functions included are clear-to-send, request-to-send and data carrier detect.

Clear-to-Send ($\overline{CTS}$)
This high-impedance, TTL-compatible input provides automatic control of the transmitting end of a communications link via the modem clear-to-send active-LOW output by inhibiting the transmit data register empty (TDRE) status bit.

Request-to-Send ($\overline{RTS}$)
The request-to-send output enables the MPU to control a peripheral or modem via the data bus. The $\overline{RTS}$ output corresponds to the state of control register bits CR_5 and CR_6. When $CR_6 = 0$ or both CR_5 and $CR_6 = 1$, the $\overline{RTS}$ output is LOW (the active state.) This output can also be used for data terminal ready ($\overline{DTR}$).

Data Carrier Detect ($\overline{DCD}$)
This high-impedance, TTL-compatible input provides automatic control, such as in the receiving end of a communications link, by means of a modem Data Carrier Detect output. The $\overline{DCD}$ input inhibits and initializes the receiver section of the ACIA when HIGH. A LOW-to-HIGH transition of $\overline{DCD}$ initiates an interrupt to the MPU to indicate the occurrence of a loss of carrier when the receive interrupt enable bit is set. The RxCLK must be running for proper $\overline{DCD}$ operation.

ACIA Registers
The block diagram for the ACIA indicates the internal registers on the chip that are used for the status, control, receiving, and transmitting of data. The content of each of the registers is summarized in *Table 1*.

Transmit Data Register (TDR)
Data is written into the transmit data register during the negative transition of the E (Enable) pulse when the ACIA has been addressed with RS HIGH and R/$\overline{W}$ LOW. Writing data into the register causes the TDRE bit in the status register to go LOW. Data can then be transmitted. If the transmitter is idling and no character is being transmitted, the transfer takes place within one bit time of the trailing edge of the write command. If a character is being transmitted, the new data character commences as soon as the previous character is complete. The transfer of data causes the TDRE bit to indicate empty.

Receive Data Register (RDR)
Data is automatically transferred to the empty receive data register (RDR) from the receiver deserializer (a shift register) upon receiving a complete character. This event causes the receive data register full (RDRF) bit in the status buffer to go HIGH (full). Data may then be read through the bus by addressing the ACIA and selecting the RDR with RS and R/$\overline{W}$ HIGH when the ACIA is enabled. The non-destructive read cycle causes the RDRF bit to be cleared to empty although the data is retained in the RDR. The status is maintained by the RDRF bit to indicate whether or not the data is current. When the receive data register is full, the automatic transfer of data from the receiver shift register to the data register is inhibited and the RDR contents remain valid, with its current status stored in the status register.

Control Register
The ACIA control register consists of eight bits of write-only buffer that are selected when RS and R/$\overline{W}$ are LOW. This register controls the function of the receiver, transmitter, interrupt enables, and the request-to-send peripheral/modem control output.

Counter Divide Select Bits (CR_0 and CR_1)
The counter divide select bits (CR_0 and CR_1) determine the divide ratios utilized in both the transmitter and receiver sections of the ACIA. Additionally, these bits are used to provide a master reset for the ACIA that clears the status register (except for external conditions on $\overline{CTS}$ and $\overline{DCD}$) and initializes both the receiver and transmitter. Master reset does not affect other control register bits. Note that after power-on or a power fail/start, these bits must be set HIGH to reset the ACIA. After resetting, the clock divide ratio may be selected.

Table 1 Definition of ACIA Register Contents

Data Bus Line Number	Buffer Address			
	RS • $\overline{R/W}$ Transmit Data Register (Write Only)	RS • R/$\overline{W}$ Receive Data Register (Read Only)	$\overline{RS}$ • $\overline{R/W}$ Control Register (Write Only)	$\overline{RS}$ • R/$\overline{W}$ Status Register (Read Only)
0	Data Bit 0*	Data Bit 0	Counter Divide Select 1 (CR_0)	Receive Data Register Full (RDRF)
1	Data Bit 1	Data Bit 1	Counter Divide Select 2 (CR_1)	Transmit Data Register Empty (TDRE)
2	Data Bit 2	Data Bit 2	Word Select 1 (CR_2)	Data Carrier Detect ($\overline{DCD}$)
3	Data Bit 3	Data Bit 3	Word Select 2 (CR_3)	Clear-to-Send ($\overline{CTS}$)
4	Data Bit 4	Data Bit 4	Word Select 3 (CR_4)	Framing Error (FE)
5	Data Bit 5	Data Bit 5	Transmit Control 1 (CR_5)	Receiver Overrun (OVRN)
6	Data Bit 6	Data Bit 6	Transmit Control 2 (CR_6)	Parity Error (PE)
7	Data Bit 7***	Data Bit 7**	Receive Interrupt Enable (CR_7)	Interrupt Request ($\overline{IRQ}$)

*Leading bit = LSB = Bit 0 **Data bit is zero in 7-bit plus parity modes. ***Data bit is "don't care" in 7-bit plus parity modes.

These counter select bits provide for the following clock divide ratios:

CR_1	CR_0	Function
0	0	÷1
0		÷16
1	0	÷64
1	1	Master Reset

Word Select Bits (CR_2, CR_3 and CR_4)
The word select bits are used to select word length, parity, and the number of stop bits. The encoding format is as follows:

CR_4	CR_3	CR_2	Function
0	0	0	7 Bits + Even Parity + 2 Stop Bits
0	0	1	7 Bits + Odd Parity + 2 Stop Bits
0	1	0	7 Bits + Even Parity + 1 Stop Bit
0	1	1	7 Bits + Odd Parity + 1 Stop Bit
1	0	0	8 Bits + 2 Stop Bits
1	0	1	8 Bits + 1 Stop Bit
1	1	0	8 Bits + Even Parity + 1 Stop Bit
1	1	1	8 Bits + Odd Parity + 1 Stop Bit

Word length, parity select, and stop bit changes are not buffered and therefore become effective immediately.

Transmitter Control Bits (CR_5 and CR_6)
Two transmitter control bits provide for the control of the interrupt from the transmit data register empty condition, the request-to-send ($\overline{RTS}$) output, and the transmission of a break level (space). The following encoding format is used:

CR_6	CR_5	Function
0	0	$\overline{RTS}$ = LOW, Transmitting Interrupt Disabled
0	1	$\overline{RTS}$ = LOW, Transmitting Interrupt Enabled
1	0	$\overline{RTS}$ = HIGH, Transmitting Interrupt Disabled
1	1	$\overline{RTS}$ = LOW, Transmits a Break Level on the Transmit Data Output. Transmitting Interrupt Disabled.

Receive Interrupt Enable Bit (CR_7)
The following interrupts are enabled by a HIGH level in bit position 7 of the control register (CR_7): receive data register full, overrun, or a LOW-to-HIGH transition on the data carrier detect ($\overline{DCD}$) signal line.

Status Register
Information on the status of the ACIA is available to the MPU by reading the ACIA status register. This read-only register is selected when RS is LOW and R/$\overline{W}$ is HIGH. Information stored in this register indicates the status of the transmit data register, the receive data register and error logic, and the peripheral/modem status inputs of the ACIA.

Receive Data Register Full (RDRF), Bit 0
Receive data register full indicates that received data has been transferred to the receive data register. The RDRF bit is cleared after an MPU read of the receive data register or by a master reset. The cleared or empty state indicates that the

contents of the receive data register are not current. Data carrier detect being HIGH also causes RDRF to indicate empty.

Transmit Data Register Empty (TDRE), Bit 1
The transmit data register empty bit being set HIGH indicates that the transmit data register contents have been transferred and that new data may be entered. The LOW state indicates that the register is full and that transmission of a new character has not begun since the last write data command.

Data Carrier Detect ($\overline{DCD}$), Bit 2
The data carrier detect bit is HIGH when the $\overline{DCD}$ input from a modem has gone HIGH to indicate that a carrier is not present. This bit going HIGH causes an interrupt request to be generated when the receive interrupt enable is set. It remains HIGH after the $\overline{DCD}$ input is returned LOW until cleared by reading first the status register and then the data register, or until a master reset occurs. If the $\overline{DCD}$ input remains HIGH after read status and read data or master reset has occurred, the interrupt is cleared, and the $\overline{DCD}$ status bit remains HIGH and will follow the $\overline{DCD}$ input.

Clear-to-Send ($\overline{CTS}$), Bit 3
The clear-to-send bit indicates the state of the clear-to-send input from a modem. A LOW $\overline{CTS}$ indicates that there is a clear-to-send from the modem. In the HIGH state, the transmit data register empty bit is inhibited and the clear-to-send status bit is HIGH. Master reset does not affect the clear-to-send status bit.

Framing Error (FE), Bit 4
Framing error indicates that the received character is improperly framed by a start and a stop bit and is detected by the absence of the first stop bit. This error indicates a synchronization error, faulty transmission, or a break condition. The framing error flag is set or reset during the receive data transfer time. Therefore, this error indicator is present throughout the time that the associated character is available.

Receiver Overrun (OVRN), Bit 5
Overrun is an error flag that indicates that one or more characters in the data stream were lost. That is, a character or a number of characters were received but not read from the receive data register (RDR) prior to subsequent characters being received. The overrun condition begins at the midpoint of the last bit of the second character received in succession without a read of the RDR having occurred. The overrun does not occur in the status register until the valid character prior to overrun has been read. The RDRF bit remains set until the overrun is reset. Character synchronization is maintained during the overrun condition. The overrun indication is reset after the reading of data from the receive data register or by a master reset.

Parity Error (PE), Bit 6
The parity error flag indicates that the number of HIGHs (1's) in the character does not agree with the preselected odd or even parity. Odd parity is defined to be when the total number of ones is odd. The parity error indication is present as long as the data character is in the RDR. If no parity is selected, both the transmitter parity generator output and the receiver parity check results are inhibited.

Interrupt Request (IRQ), Bit 7
The IRQ bit indicates the state of the $\overline{IRQ}$ output. Any interrupt condition with its applicable enable is indicated in this status bit. Any time the $\overline{IRQ}$ output is LOW, the IRQ bit is HIGH to indicate the interrupt or service request status. The IRQ bit is cleared by a read operation to the receive data register or a write operation to the transmit data register.

Absolute Maximum Ratings

Supply Voltage	−0.3 V, +7.0 V
Input Voltage	−0.3 V, +7.0 V
Operating Temperature − T_L, T_H	
F6850, F68A50, F68B50	0°C, +70°C
F6850C, F68A50C	−40°C, +85°C
F6850DL	−55°C, +85°C
F6850DM	−55°C, +125°C
Storage Temperature Range	−55°C, +150°C
Thermal Resistance	
Ceramic	60°C/W
Plastic	120°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit.

DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted

Symbol	Characteristic	Signal	Min	Typ	Max	Unit	Condition
V_{IH}	Input HIGH Voltage		2.0		V_{CC}	V	
V_{IL}	Input LOW Voltage		−0.3		0.8	V	
I_{IN}	Input Leakage Current	R/$\overline{W}$, CS_0, CS_1, $\overline{CS_2}$, RS, Rx DATA, Rx CLK, $\overline{CTS}$, $\overline{DCD}$		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	3-State Input Current (OFF State)	D_0-D_7		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
V_{OH}	Output HIGH Voltage	D_0-D_7	2.4			V	I_{Load} = −205 μA, Enable Pulse Width < 25 μs
		Tx DATA, $\overline{RTS}$	2.4				I_{Load} = −100 μA, Enable Pulse Width < 25 μs
V_{OL}	Output LOW Voltage				0.4	V	I_{Load} = 1.6 mA, Enable Pulse Width < 25 μs
I_{LOH}	Output Leakage Current	$\overline{IRQ}$		1.0	10	μA	V_{OH} = 2.4 V
P_D	Power Dissipation			300	525	mW	
C_{IN}	Input Capacitance	D_0-D_7		10	12.5	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
		E, Tx CLK, Rx CLK, R/$\overline{W}$, RS, Rx DATA, CS_0, CS_1, $\overline{CS_2}$, $\overline{CTS}$, $\overline{DCD}$		7.0	7.5		
C_{OUT}	Output Capacitance	$\overline{RTS}$, Tx DATA			10	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
		$\overline{IRQ}$			5.0		

5

AC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted

Symbol	Characteristic	F6850		F68A50		F68B50		Unit	Condition
		Min	Max	Min	Max	Min	Max		
PW_{CL}	Minimum Clock Pulse Width, LOW ÷16, ÷64 Modes	600		450		280		ns	*Figure 1*
PW_{CH}	Minimum Clock Pulse Width, HIGH ÷16, ÷64 Modes	600		450		280		ns	*Figure 2*
f_C	Clock Frequency ÷1 Mode ÷16, ÷64 Modes		500 800		750 1000		1000 1500	kHz	
t_{TDD}	Clock-to-Data Delay for Transmitter		600		540		460	ns	*Figure 3*
t_{RDS}	Receive Data Set-up Time ÷1 Mode	250		100		30		ns	*Figure 4*
t_{RDH}	Receive Data Hold Time ÷1 Mode	250		100		30		ns	*Figure 5*
t_{IR}	Interrupt Request Release Time		1.2		0.9		0.7	μs	*Figure 6*
t_{RTS}	Request-to-Send Delay Time		560		480		400	ns	*Figure 6*
t_r, t_f	Input Transition Times (Except Enable)		1.0		0.5		0.25	μs	Note

Note
1.0 μs or 10% of the pulse width, whichever is smaller

Bus Timing Characteristics

Read *(Figures 7 and 9)*

Symbol	Characteristic	F6850		F68A50		F68B50		Unit
		Min	Max	Min	Max	Min	Max	
$t_{cyc}E$	Enable Cycle Time	1.0		0.666		0.500		μs
PW_{EH}	Enable Pulse Width, HIGH	0.45	25	0.28	25	0.22	25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.28		0.21		μs
t_{AS}	Set-up Time, Address, and R/$\overline{W}$ Valid to Enable Positive Transition	160		140		70		ns
t_{DDR}	Data Delay Time		320		220		180	ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input		25		25		25	ns
Write *(Figures 8 and 9)*								
$t_{cyc}E$	Enable Cycle Time	1.0		0.666		0.500		μs
PW_{EH}	Enable Pulse Width, HIGH	0.45	25	0.28	25	0.22	25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.28		0.21		μs
t_{AS}	Set-up Time, Address, and R/$\overline{W}$ Valid to Enable Positive Transition	160		140		70		ns
t_{DSW}	Data Set-up Time	100		80		60		ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input		25		25		25	ns

Fig. 1 Clock Pulse Width, LOW State

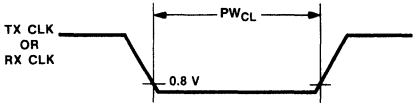


Fig. 2 Clock Pulse Width, HIGH State

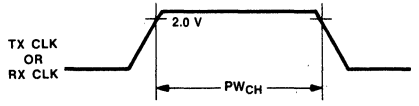


Fig. 3 Transmit Data Output Delay

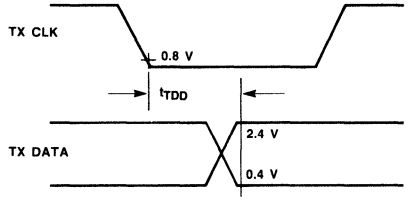


Fig. 4 Receive Data Set-up Time (÷1 Mode)

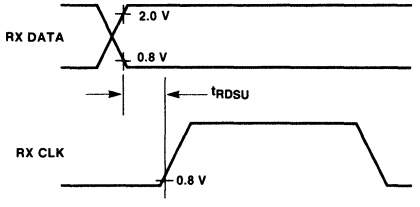


Fig. 5 Receive Data Hold Time (÷1 Mode)

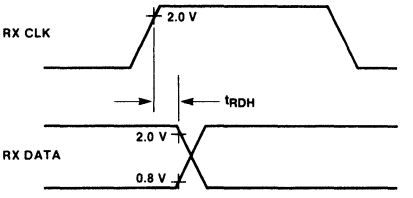


Fig. 6 Request-to-Send Delay and Interrupt Request Release Times

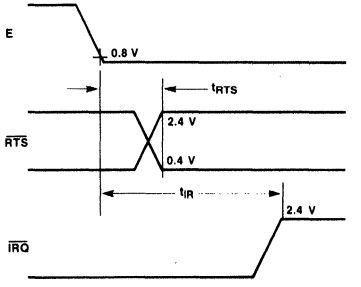
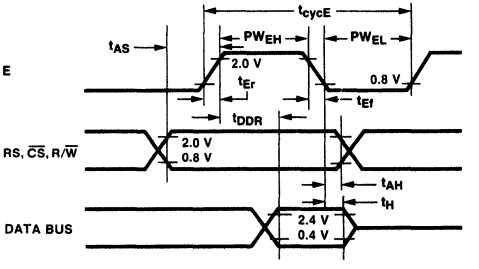


Fig. 7 Bus Read Timing Characteristics (Read Information from ACIA)



5

Fig. 8 Bus Write Timing Characteristics (Write Information into ACIA)

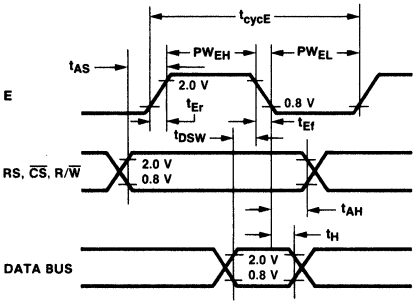
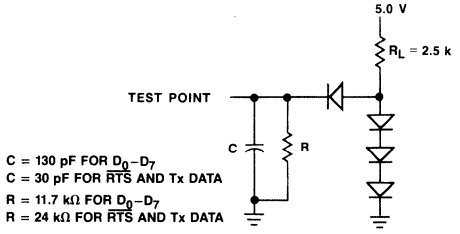
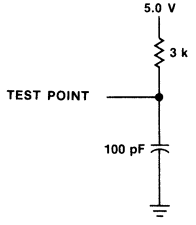


Fig. 9 Bus Timing Test Loads

Load A (D_0–D_7, $\overline{RTS}$, Tx DATA)



Load B ($\overline{IRQ}$ Only)



Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6850P,S	0°C to 70°C
	F6850CP,CS	−40°C to +85°C
	F6850DL	−55°C to +85°C
	F6850DM	−55°C to +125°C
1.5 MHz	F68A50P,S	0°C to +70°C
	F68A50CP,CS	−40°C to +85°C
2.0 MHz	F68B50DM	−55°C to +125°C
	F68B50P,S	0°C to +70°C

P = Plastic package, S = Ceramic package

FAIRCHILD
A Schlumberger Company

F6852/F68A52/F68B52 Synchronous Serial Data Adapter

Microprocessor Product

Description

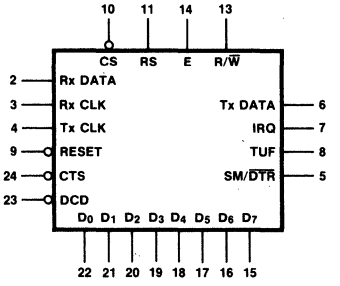
The F6852 Synchronous Serial Data Adapter (SSDA) provides a bidirectional serial interface for synchronous data information interchange. It contains interface logic for simultaneously transmitting and receiving standard synchronous communications characters in bus-organized systems, such as the F6800 microprocessor systems.

The bus interface of the F6852 includes Select, Enable, Read/Write, Interrupt, and bus interface logic to allow data transfer over an 8-bit bidirectional data bus. The parallel data of the bus system is transmitted serially and received by the synchronous data interface with synchronization, fill character insertion/deletion, and error checking. The functional configuration of the SSDA is programmed via the data bus during system initialization. Programmable control registers provide control for variable word lengths, transmit control, receive control, synchronization control, and interrupt control. Status, timing, and control lines provide peripheral or modem control.

Typical applications include floppy disk controllers, cassette or cartridge tape controllers, data communications terminals, and numerical control systems.

- **Programmable Interrupts from Transmitter, Receiver, and Error Detection Logic**
- **Character Synchronization on 1 or 2 SYNC Codes**
- **External Synchronization Available for Parallel-Serial Operation**
- **Available Speeds: 1.0 MHz for the F6852, 1.5 MHz for the F68A52, and 2.0 MHz for the F68B52**
- **Programmable SYNC Code Register**
- **Up to 600K BPS Transmission**
- **Peripheral/Modem Control Functions**
- **3 Bytes of FIFO Buffering on Both Transmit and Receive**
- **7-, 8-, or 9-Bit Transmission**
- **Optional Even and Odd Parity**
- **Parity, Overrun, and Underflow Status**

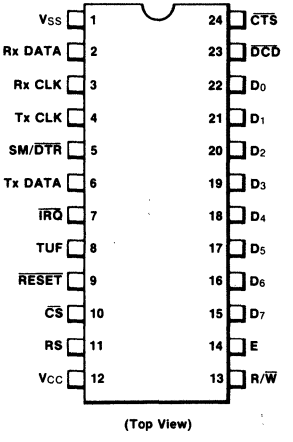
Logic Symbol



V_{SS} = Pin 1
V_{CC} = Pin 12

5

Connection Diagram 24-Pin DIP

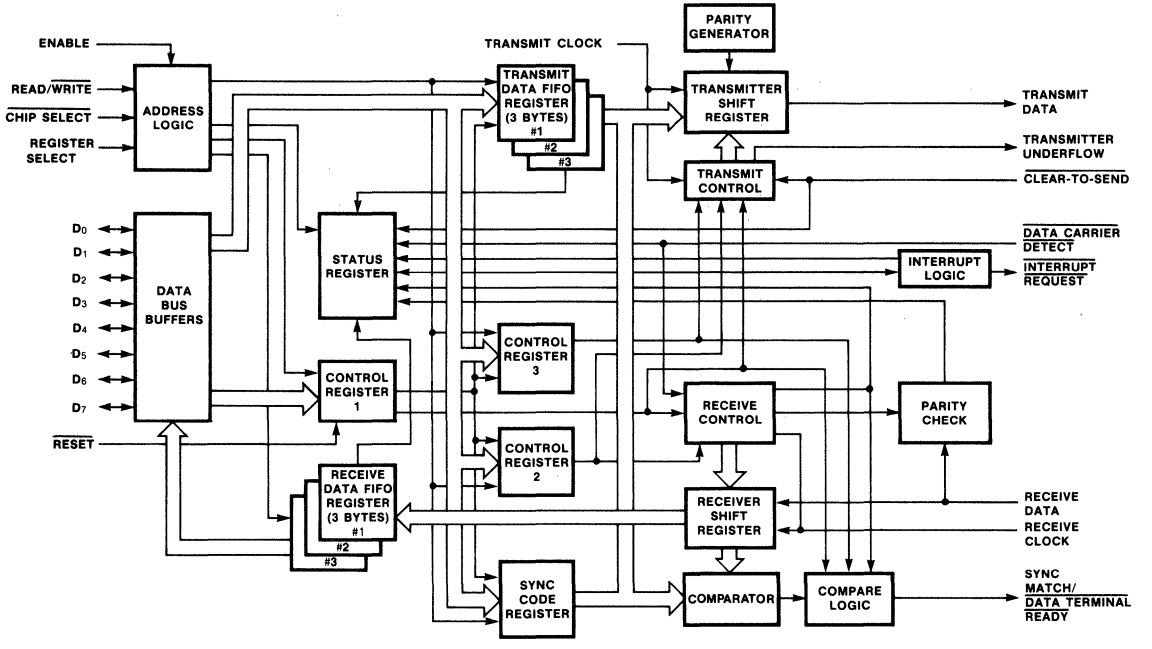


(Top View)

Pin Names

Rx DATA	Receive Data Input
Rx CLK	Receive Clock Input
Tx CLK	Transmit Clock Input
SM/DTR	Sync Match/Data Terminal Ready Output
Tx DATA	Transmit Data Output
IRQ	Interrupt Request Output
TUF	Transmitter Underflow Output
RESET	Reset Input
CS	Chip Select Input
RS	Register Select Input
CTS	Clear-to-Send Input
DCD	Data Carrier Detect Input
D_0-D_7	Bidirectional Data Lines
E	Enable (System $\phi 2$ Clock) Input
R/W	Read/Write Input
V_{SS}	Ground Input
V_{CC}	+5 V Power Supply Input

Block Diagram



Device Operation
At the bus interface, the SSDA appears as two addressable memory locations. Internally, there are seven registers: two read-only and five write-only registers. The read-only registers are status and receive data; the write-only registers are control 1, control 2, control 3, sync code, and transmit data. The serial interface consists of serial input and output lines with independent clocks, and four peripheral/modem control lines.

Data to be transmitted is transferred directly into the 3-byte transmit data first-in, first-out (FIFO) register from the data bus. Availability of the input to the FIFO is indicated by a bit in the status register; once data is entered, it moves through the FIFO to the last empty location. Data at the output of the FIFO is automatically transferred from the FIFO to the transmitter shift register as the shift register becomes available to transmit the next character. If data is not available from the FIFO (underflow condition), the transmitter shift register is automatically loaded with either a sync code or an all-"1s" character. The transmit section may be programmed to append even, odd, or no parity to the transmitted word. An external Clear-to-Send ($\overline{CTS}$) control line is provided to inhibit the transmitter without clearing the FIFO.

Serial data is accumulated in the receiver based on the synchronization mode selected. In the external sync mode, used for parallel-serial operation, the receiver is synchronized by the Data Carrier Detect ($\overline{DCD}$) input and transfers successive bytes of data to the input of the receiver FIFO. The single-sync-character mode requires that a match occur between the sync code register and one incoming character before data transfer to the FIFO begins. The two-sync-character mode requires that two sync codes be received in sequence to establish synchronization. Subsequent to synchronization in any mode, data is accumulated in the shift register and parity is optionally checked. An indication of parity error is carried through the receiver FIFO with each character to the last empty location. Availability of a word at the FIFO output is indicated by a bit in the status register, as is a parity error.

The SSDA and its internal registers are selected by the address bus, Read/Write (R/$\overline{W}$), and Enable control lines. To configure the SSDA, control registers are selected and the appropriate bits set. The status register is addressable for reading status.

Other I/O lines, in addition to Clear-to-Send ($\overline{CTS}$) and Data Carrier Detect ($\overline{DCD}$), include Sync Match/Data Terminal Ready (SM/$\overline{DTR}$) and Transmitter Underflow (TUF). The transmitter and receiver each have individual clock inputs, allowing simultaneous operation under separate clock control. Signals to the microprocessor are the data bus and Interrupt Request ($\overline{IRQ}$).

Initialization
During a power-on sequence, the SSDA is reset via the $\overline{RESET}$ input and internally latched in a reset condition to prevent erroneous output transitions. The sync code register, control register 2, and control register 3 should be programmed prior to the programmed release of the transmitter and/or receiver reset bits; these bits in control register 1 should be cleared after the $\overline{RESET}$ line has gone HIGH.

Transmitter Operation
Data is transferred to the transmitter section in parallel form by means of the data bus and transmit data FIFO. The transmit data FIFO is a 3-byte register whose status is indicated by the transmitter data register available (TDRA) status bit and its associated interrupt enable bit. Data is transferred through the FIFO on negative edges of Enable (E) pulses. Two data transfer modes are provided in the SSDA. The 1-byte transfer mode provides for writing data to the transmitter section (and reading from the receiver section) one byte at a time. The 2-byte transfer mode provides for writing two data characters in succession.

Data will automatically transfer from the last register location in the transmit data FIFO (when it contains data) to the transmitter shift register during the last half of the last bit of the previous character. A character is transferred into the shift register by the Transmit Clock. Data is transmitted *LSB first*, and odd or even parity can be optionally appended. The unused bit positions in short word length characters from the data bus are "don't cares". (Note: The data bus inputs may be reversed for applications requiring the MSB to be transferred first, e.g., IBM format for floppy disks; however, care must be taken to program the control registers properly — *Table 1* will have its bit positions reversed.)

When the shift register becomes empty, and data is *not* available for transfer from the transmit data FIFO, an underflow occurs, and a character is inserted into the transmitter data stream to maintain character synchronization. The character transmitted on underflow will be either a mark (all "1s") or the contents of the sync code register, depending upon the state of the transmit sync code on underflow control bit. The underflow condition is indicated by a pulse (≈ Tx CLK HIGH period) on the Transmitter Underflow output (when in Tx Sync on underflow mode). The Transmitter Underflow output

occurs coincident with the transfer of the last half of the last bit preceding the underflow character. The underflow status bit is set until cleared by means of the clear underflow control bit. This output may be used in floppy disk systems to synchronize write operations and for appending CRCC.

Transmission is initiated by clearing the transmitter reset bit in control register 1. When the transmitter reset bit is cleared, the first *full* positive half-cycle of the Transmit Clock will initiate the transmit cycle, with the transmission of data or underflow characters beginning on the negative edge of the Transmit Clock pulse that started the cycle. If the transmit data FIFO was not loaded, an underflow character will be transmitted (see *Figure 4*).

The Clear-to-Send ($\overline{CTS}$) input provides for automatic control of the transmitter by means of external system hardware; e.g., the modem $\overline{CTS}$ output provides the control in a data communications system. The $\overline{CTS}$ input resets and inhibits the transmitter section when HIGH, but does not reset the transmit data FIFO. The TDRA status bit is inhibited by $\overline{CTS}$ being HIGH in either the one-sync-character or two-sync-character mode of operation. In the external sync mode, TDRA is unaffected by $\overline{CTS}$ in order to provide transmit data FIFO status for preloading and operating the transmitter under the control of the $\overline{CTS}$ input. When the transmitter reset bit (Tx Rs) is set, the transmit data FIFO is cleared and the TDRA status bit is cleared. After one E clock has occurred, the transmit data FIFO becomes available for new data with TDRA inhibited.

Receiver Operation

Data and a presynchronized clock are provided to the SSDA receiver section by means of the Receive Data (Rx DATA) and Receive Clock (Rx CLK) inputs. The data is a continuous stream of binary data bits without means for identifying character boundaries within the stream. It is, therefore, necessary to achieve character synchronization for the data at the *beginning* of the data block. Once synchronization is achieved, it is assumed to be retained for all successive characters within the block.

Data communication systems utilize the detection of sync codes during the initial portion of the preamble to establish character synchronization. This requires the detection of a single code or two successive sync codes. Floppy disk and cartridge tape units require 16 bits of defined preamble and cassettes require eight bits of preamble to establish the reference for the start of record. All three are functionally equivalent to the detection of sync codes. Systems that do not utilize code detection techniques require custom logic external to the SSDA for character synchronization and use of the parallel-to-serial (external sync) mode. (Note: The receiver shift register is set to "1s" when reset.)

Synchronization

The SSDA provides three operating modes with respect to character synchronization: one-sync-character mode, two-sync-character mode, and external sync mode. The external sync mode requires synchronization and control of the receiving section through the Data Carrier Detect ($\overline{DCD}$) input (see *Figure 7*). This external synchronization could consist of direct line control from the transmitting end of the serial data link or from external logic designed to detect the start of the message block. The one-sync-character mode searches on a bit-by-bit basis until a match is achieved between the data in the shift register and the sync code register. The match indicates character synchronization is complete and will be retained for message block. In the two-sync-character mode, the receiver searches for the first sync code match on a bit-by-bit basis and then looks for a second *successive* sync code character prior to establishing character synchronization. If the second sync code character is not received, the bit-by-bit search for the first sync code is resumed.

Sync codes received prior to the completion of synchronization (one or two characters) are not transferred to the receive data FIFO. Redundant sync codes during the preamble or sync codes that occur as "fill characters" can be automatically stripped from the data, when the strip sync control bit is set, to minimize system loading. The character synchronization will be retained until cleared by means of the clear sync bit, which also inhibits synchronization search when set.

Receiving Data

Once synchronization has been achieved, subsequent characters are automatically transferred into the receive data FIFO and clocked through the FIFO to the last empty location by E pulses (MPU system $\phi 2$). The receiver data available (RDA) status bit indicates when data is available to be read from the last FIFO location (No. 3) when in the 1-byte transfer mode. The 2-byte transfer mode causes the RDA status bit to indicate data is available when the last two FIFO register locations are full. Data being available in the receive data FIFO causes an interrupt request if the receiver interrupt enable (RIE) bit is set. The MPU will then read the SSDA status register, which will indicate that data is available for the MPU read from the receive data FIFO register. The IRQ and RDA status bits are reset by a read from the FIFO. If more than one character has been received and is resident in the receive data FIFO,

subsequent E clocks will cause the FIFO to update, and the RDA and IRQ status bits will be set again. The read data operation for the 2-byte transfer mode requires an intervening E clock between reads to allow the FIFO data to shift. Optional parity is automatically checked as data is received, and the parity status condition is maintained with each character until the data is read from the receive data FIFO. Parity errors will cause an interrupt request if the error interrupt enable (EIE) has been set. The parity bit is not transferred to the data bus but must be checked in the status register. Note: In the 2-byte transfer mode, parity should be checked prior to reading the second byte, since a FIFO read clears the error bit.

Other status bits that pertain to the receiver section are receiver overrun and data carrier detect ($\overline{DCD}$). The overrun status bit is automatically set when a transfer of a character to the receive data FIFO occurs and the first register of the receive data FIFO is full. Overrun causes an interrupt if error interrupt enable (EIE) has been set. The transfer of the overrunning character into the FIFO causes the previous character in the FIFO input register location to be lost. The overrun status bit is cleared by reading the status register (when the overrun condition is present), followed by a receive data FIFO register read. Overrun cannot occur and be cleared without providing an opportunity to detect the occurrence via the status register.

A positive transition on the $\overline{DCD}$ input causes an interrupt if the EIE control bit has been set. The interrupt caused by $\overline{DCD}$ is cleared by reading the status register when the $\overline{DCD}$ status bit is HIGH, followed by a receive data FIFO read. The $\overline{DCD}$ status bit will subsequently follow the state of the $\overline{DCD}$ input when it goes LOW.

SSDA Interface Signals for MPU

The SSDA interfaces to the F6800 MPU with an 8-bit bidirectional data bus, a Chip Select line, a Register Select line, an Interrupt Request line, a Read/Write line, an Enable line, and a Reset line. These signals, in conjunction with the F6800 VMA output, permit the MPU to have complete control over the SSDA.

Bidirectional Data Bus (D_0-D_7)
The bidirectional data lines (D_0-D_7) allow for data transfer between the SSDA and the MPU. The data bus output drivers are 3-state devices that remain in the high-impedance (OFF) state except when the MPU performs an SSDA read operation.

Enable (E)
The Enable (E) signal is a high-impedance, TTL-compatible input that enables the bus input/output data buffers, clocks data to and from the SSDA, and moves data through the FIFO registers. This signal is normally the continuous F6800 system $\phi 2$ clock, so that incoming data characters are shifted through the FIFO.

Read/Write (R/$\overline{W}$)
The Read/Write line is a high-impedance input that is TTL-compatible and is used to control the direction of data flow through the SSDAs input/output data bus interface. When Read/Write is HIGH (MPU read cycle), SSDA output drivers are turned on if the device is selected and a selected register is read. When it is LOW, the SSDA output drivers are turned off and the MPU writes into a selected register. The Read/Write signal is also used to select read-only or write-only registers within the SSDA.

Chip Select ($\overline{CS}$)
This high-impedance, TTL-compatible input line is used to address the SSDA. The SSDA is selected when $\overline{CS}$ is LOW. VMA should be used in generating the $\overline{CS}$ input to insure that false selects will not occur. Transfers of data to and from the SSDA are then performed under the control of the Enable signal, Read/Write, and Register Select.

Register Select (RS)
The Register Select line is a high-impedance input that is TTL-compatible. A HIGH level is used to select control registers C2 and C3, the sync code register, and the transmit/receive data registers. A LOW level selects the control 1 and status registers (see *Table 1*).

Interrupt Request ($\overline{IRQ}$)
Interupt Request is a TTL-compatible, open-drain (no internal pull-up), active-LOW output that is used to interrupt the MPU. The Interrupt Request remains LOW until cleared by the MPU.

Reset Input ($\overline{RESET}$)
The Reset input provides a means of resetting the SSDA from an external source. In the LOW state, the Reset input causes the following:

1. Receiver reset (Rx Rs) and transmitter reset (Tx Rs) bits are set, causing both the receiver and transmitter sections to be held in a reset condition.
2. Peripheral control bits PC1 and PC2 are reset to "0", causing the SM/$\overline{DTR}$ output to be HIGH.
3. The error interrupt enable (EIE) bit is reset.
4. An internal synchronization mode is selected.
5. The transmitter data register available (TDRA) status bit is cleared and inhibited.

When $\overline{\text{RESET}}$ returns HIGH (the inactive state), the transmitter and receiver sections will remain in the reset state until the receiver reset and transmitter reset bits are cleared via the bus under software control. The control register bit affected by $\overline{\text{RESET}}$ (Rx Rs, Tx Rs, PC1, PC2, EIE, and E/L Sync) cannot be changed when $\overline{\text{RESET}}$ is LOW.

Clock Inputs

Separate high-impedance, TTL-compatible inputs are provided for clocking of transmitted and received data.

Transmit Clock (Tx CLK)
The Transmit Clock input is used for the clocking of transmitted data. The transmitter shifts data on the negative transition of the clock.

Receive Clock (Rx CLK)
The Receive Clock input is used for clocking in received data. The clock and data must be synchronized externally. The receiver samples the data on the positive transition of the clock.

Serial Input/Output Lines

Receive Data (Rx DATA)
The Receive Data line is a high-impedance, TTL-compatible input through which data is received in a serial format. Data rates are from 0 to 600K bps.

Transmit Data (Tx DATA)
The Transmit Data output line transfers serial data to a modem or other peripheral. Data rates are from 0 to 600K bps.

Peripheral/Modem Control

The SSDA includes several functions that permit limited control of a peripheral or modem. The functions included are Clear-to-Send, Sync Match/Data Terminal Ready, Data Carrier Detect, and Transmitter Underflow.

Clear-to-Send ($\overline{\text{CTS}}$)
The $\overline{\text{CTS}}$ input provides a real-time inhibit to the transmitter section (the transmit data FIFO is not disturbed). A positive $\overline{\text{CTS}}$ transition resets the transmitter shift register and inhibits the TDRA status bit and its associated interrupt in both the one-sync-character and two-sync-character modes of operation. TDRA is not affected by the $\overline{\text{CTS}}$ input in the external sync mode.

The positive transition of $\overline{\text{CTS}}$ is stored within the SSDA to insure that its occurrence will be acknowledged by the system. The stored $\overline{\text{CTS}}$ information and its associated $\overline{\text{IRQ}}$ (if enabled) are cleared by writing a "1" in the $\overline{\text{CTS}}$ bit in control register 3 or in the transmitter reset bit. The $\overline{\text{CTS}}$ status bit subsequently follows the $\overline{\text{CTS}}$ input when it goes LOW.

The $\overline{\text{CTS}}$ input provides character timing for transmitter data when in the external sync mode. Transmission is initiated on the negative transition of the first *full* positive clock pulse of the Transmit Clock (Tx CLK) after the release of $\overline{\text{CTS}}$. See *Figure 6*.

Data Carrier Detect ($\overline{\text{DCD}}$)
The $\overline{\text{DCD}}$ input provides a real-time inhibit to the receiver section (the Rx FIFO is not disturbed). A positive $\overline{\text{DCD}}$ transition resets and inhibits the receiver section except for the receive FIFO and the RDRA status bit and its associated $\overline{\text{IRQ}}$.

The positive transition of $\overline{\text{DCD}}$ is stored within the SSDA to insure that its occurrence will be acknowledged by the system. The stored $\overline{\text{DCD}}$ information and its associated $\overline{\text{IRQ}}$ (if enabled) are cleared by reading the status register and then the receive FIFO, or by writing a "1" into the receiver reset bit. The $\overline{\text{DCD}}$ status bit subsequently follows the $\overline{\text{DCD}}$ input when it goes LOW. The $\overline{\text{DCD}}$ input provides character synchronization timing for the receiver during the external sync mode of operation. The receiver will be initialized and data will be sampled on the positive transition of the first *full* Receive Clock cycle after release of $\overline{\text{DCD}}$. See *Figure 7*.

Sync Match/Data Terminal Ready (SM/$\overline{\text{DTR}}$)
The SM/$\overline{\text{DTR}}$ output provides four functions depending on the state of the PC1 and PC2 control bits. When the Sync Match mode is selected (PC1 = "1", PC2 = "0"), the output provides a one-bit-wide pulse when a sync code is detected. This pulse occurs for each sync code match even if the receiver has already attained synchronization. The SM output is inhibited when PC2 = "1". The $\overline{\text{DTR}}$ mode (PC1 = "0") provides an output level corresponding to the complement of PC2 ($\overline{\text{DTR}}$ = "0" when PC2 = "1"). See *Table 1*.

Transmitter Underflow (TUF)
The Transmitter Underflow output indicates the occurrence of a transfer of a "fill character" to the transmitter shift register when the last location (No. 3) in the transmit data FIFO is empty. The Transmitter Underflow output pulse is approximately a Tx CLK HIGH period wide and occurs during the last half of the last bit of the character preceding the underflow. See *Figure 4*. The Transmitter Underflow output pulse does not occur when the Tx Sync bit is in the reset state.

SSDA Registers
Seven registers in the SSDA can be accessed by means of the bus. The registers are defined as a read-only or write-only according to the direction of information flow. The Register Select input (RS) selects two registers in each state, one being read-only and the other write-only. The Read/Write input (R/$\overline{W}$) defines which of the two selected registers will actually be accessed. Four registers (two read-only and two write-only) can be addressed via the bus at any particular time. These registers and the required addressing are defined in *Table 1*.

Control Register 1 (C1)

Control register 1 is an 8-bit, write-only register that can be addressed directly from the data bus. Control register 1 is addressed when RS = "0" and R/$\overline{W}$ = "0".

Receiver Reset (Rx Rs), C1 Bit 0
The receiver reset control bit provides both a reset and inhibit function to the receiver section. When Rx Rs is set, it clears the receiver control logic, sync logic, error logic, Rx Data FIFO control, parity error status bit, and $\overline{DCD}$ interrupt. The receiver shift register is set to "1s". The Rx Rs bit must be cleared after the occurrence of a LOW level on $\overline{RESET}$ in order to enable the receiver section of the SSDA.

Transmitter Reset (Tx Rs), C1 Bit 1
The transmitter reset control bit provides both a reset and inhibit to the transmitter section. When Tx Rs is set, it clears the transmitter control section, transmitter shift register, Tx Data FIFO (which can be reloaded after one E clock pulse), the transmitter underflow status bit, and the $\overline{CTS}$ interrupt, and inhibits the TDRA status bit (in the one-sync-character and two-sync-character modes). The Tx Rs bit must be cleared after the occurrence of a LOW level on $\overline{RESET}$ in order to enable the transmitter section of the SSDA. If the Tx FIFO is not preloaded, it must be loaded immediately after the Tx Rs release to prevent a transmitter underflow condition.

Strip Synchronization Characters (Strip Sync), C1 Bit 2
If the strip sync bit is set, the SSDA will automatically strip all received characters that match the contents of the sync code register. The characters used for synchronization (one or two characters of sync) are always stripped from the received data stream.

Clear Synchronization (Clear Sync), C1 Bit 3
The clear sync control bit provides the capability of dropping receiver character synchronization and inhibiting resynchronization. The clear sync bit is set to clear and inhibit receiver synchronization in *all* modes and is reset to "0" to enable resynchronization.

Transmitter Interrupt Enable (TIE), C1 Bit 4
TIE enables both the Interrupt Request output ($\overline{IRQ}$) and interrupt request status bit to indicate a transmitter service request. When TIE is set and the TDRA status bit is HIGH, the $\overline{IRQ}$ output will go LOW (the active state) and the IRQ status bit will go HIGH.

Receiver Interrupt Enable (RIE), C1 Bit 5
RIE enables both the Interrupt Request output ($\overline{IRQ}$) and the interrupt request status bit to indicate a receiver service request. When RIE is set and the RDA status is HIGH, the $\overline{IRQ}$ output will go LOW (the active state) and the IRQ status bit will go HIGH.

Address Control 1 (AC1) and Address Control 2 (AC2), C1 Bits 6 and 7
AC1 and AC2 select one of the write-only registers — control 2, control 3, sync code, or transmit data FIFO — as shown in *Table 1*, when RS = "1" and R/$\overline{W}$ = "0".

Control Register 2 (C2)

Control register 2 is an 8-bit, write-only register that can be programmed from the bus when the address control bits in control register 1 (AC1 and AC2) are reset, RS = "1" and R/$\overline{W}$ = "0".

Peripheral Control 1 (PC1) and Peripheral Control 2 (PC2), C2 Bits 0 and 1
Two control bits, PC1 and PC2, determine the operating characteristics of the Sync Match/$\overline{DTR}$ output. PC1, when HIGH, selects the Sync Match mode. PC2 provides the inhibit/enable control for the SM/$\overline{DTR}$ output in the sync match mode. A one-bit-wide pulse is generated at the output when PC2 is "0", and a match occurs between the contents of the sync code register and the incoming data even if sync is inhibited (Clear Sync bit = "1".) The sync match pulse is referenced to the negative edge of Rx CLK pulse causing the match See *Figure 3.*

The Data Terminal Ready ($\overline{DTR}$) mode is selected when PC1 is LOW. When PC2 = "1", the SM/$\overline{DTR}$ output = "0", and vice versa. The operation of PC2 and PC1 is summarized in *Table 1.*

1-Byte/2-Byte Transfer (1-Byte/2-Byte), C2 Bit 2
When 1-Byte/2-Byte is set, the TDRA and RDA status bits will indicate the availability of their respective data FIFO register for a single-byte data transfer. Alternately, if 1-Byte/2-Byte is reset, the TDRA and RDA status bits indicate when two bytes of data can be moved without a second status read. An intervening Enable pulse must occur between data transfers.

Word Length Selects (WS1, WS2, WS3), C2 Bits 3, 4, 5
Word length select bits WS1, WS2, and WS3 select word length of seven, eight, or nine bits, including parity, as shown in *Table 1.*

Transmit Sync Code on Underflow (Tx Sync), C2 Bit 6
When Tx Sync is set, the transmitter will automatically send a sync character when data is not available for transmission. If Tx Sync is reset, the transmitter will transmit a mark character (including the parity bit position) on underflow. When the underflow is detected, a pulse approximately a Tx CLK HIGH period wide will occur on the underflow output if the Tx Sync bit is set. Internal parity generation is inhibited during underflow except for sync code fill character transmission in 8-bit plus parity word lengths.

Error Interrupt Enable (EIE), C2 Bit 7
When EIE is set, the IRQ status bit will go HIGH and the $\overline{IRQ}$ output will go LOW if:

1. A receiver overrun occurs. The interrupt is cleared by reading the status register and reading the Rx Data FIFO.
2. $\overline{DCD}$ input has gone to a "1". The interrupt is cleared by reading the status register and reading the Rx Data FIFO.
3. A parity error exists for the character in the last location (No. 3) of the Rx Data FIFO. The interrupt is cleared by reading the Rx Data FIFO.
4. The $\overline{CTS}$ input has gone to a "1". The interrupt is cleared by writing a "1" in the Clear $\overline{CTS}$ bit, C3 bit 2, or by Tx Reset.
5. The transmitter has underflowed (in the Tx Sync on underflow mode). The interrupt is cleared by writing a "1" into the clear underflow, C3 bit 3, or Tx Reset.

When EIE is a "0", the IRQ status bit and the $\overline{IRQ}$ output are disabled for the above error conditions. A LOW level on the $\overline{RESET}$ input resets EIE to "0".

Control Register 3 (C3)

Control Register 3 is a 4-bit, write-only register that can be programmmed from the bus when RS = "1" and R/$\overline{W}$ = "0", and address control bit AC1 = "1" and AC2 = "0".

External/Internal Sync Mode Control (E/I Sync), C3 Bit 0
When the E/I sync mode bit is HIGH, the SSDA is in the external sync mode and the receiver synchronization logic is disabled. Synchronization can be achieved by means of the DCD input or by starting Rx CLK at the midpoint of data bit 0 of a character with $\overline{DCD}$ LOW. Both the transmitter and receiver sections operate as parallel—serial converters in the external sync mode. The clear sync bit in control register 1 acts as a receiver sync inhibit when HIGH to provide a bus-controllable inhibit. The sync code register can serve as a transmitter fill character register and a receiver match register in this mode. A LOW on the $\overline{RESET}$ input resets the E/I sync mode bit, placing the SSDA in the internal sync mode.

One-Sync-Character/Two-Sync-Character Mode Control (1-Sync/2-Sync), C3 Bit 1
When the 1-Sync/2-Sync bit is set, the SSDA will synchronize on a single match between the received data and the contents of the sync code register. When the 1-Sync/2-Sync bit is reset, two successive sync characters must be received prior to receiver synchronization. If the second sync character is not detected, the bit-by-bit search resumes from the first bit in the second character. See the description of the sync code register for more details.

Clear $\overline{CTS}$ Status (Clear $\overline{CTS}$), C3 Bit 2
When a "1" is written into the $\overline{CTS}$ bit, the stored status and interrupt are cleared. Subsequently, the $\overline{CTS}$ status bit reflects the state of the $\overline{CTS}$ input. The Clear $\overline{CTS}$ control bit does not affect the $\overline{CTS}$ input nor its inhibit of the transmitter section. The Clear $\overline{CTS}$ command bit is self-clearing, and writing "0" into this bit is a nonfunctional operation.

Clear Transmit Underflow Status (CTUF), C3 Bit 3
When a "1" is written into the CTUF status bit, the CTUF bit and its associated interrupt are reset. The CTUF command bit is self-clearing and writing a "0" into this bit is a nonfunctional operation.

Sync Code Register

The sync code register is an 8-bit register for storing the programmable sync code required for received data character synchronization in the one-sync-character and two-sync-character modes. The sync code register also provides for stripping the sync/fill characters from the received data (a programmable option) as well as automatic insertion of fill characters in the transmitted data stream. The sync code register is not utilized for receiver character synchronization in the external sync mode; however, it provides storage of receiver match and transmit fill characters.

The sync code register can be loaded when AC2 and AC1 are a "1" and "0", respectively, and R/$\overline{W}$ = "0" and RS = "1".

The sync code register may be changed after the detection of a match with the received data (the first sync code having been detected) to synchronize with a double-word sync pattern. (This sync code change must occur prior to the completion of the second character.) The sync match (SM) output can be used to interrupt the MPU system to indicate that the first eight bits have matched. The service routine would then change the sync match register to the second half of the pattern. Alternately, the one-sync-character mode can be used for sync codes for 16 or more bits by using software to check the second and subsequent bytes after reading them from the FIFO.

The detection of the sync code can be programmed to appear on the Sync Match/$\overline{DTR}$ output by writing a "1" in PC1 (C2 bit 0) and a "0" in PC2 (C2 bit 1). The Sync Match output will go HIGH for one bit time beginning at the character interface between the sync code and the next character (see *Figure 3*).

Parity for Sync Character

Transmitter
The transmitter does not generate parity for the sync character except in the 9-bit mode.

9-bit (8-bit + parity) . . . 8-bit sync character + parity
8-bit (7-bit + parity) . . . 8-bit sync character (no parity)
7-bit (6-bit + parity) . . . 8-bit sync character (no parity)

Receiver

At Synchronization
The receiver automatically strips the sync character(s) (two sync characters if 2-sync mode is selected) that is used to establish synchronization. Parity is not checked for these sync characters.

After Synchronization is Established
When strip-sync bit is selected, the sync characters (fill characters) are stripped and parity is not checked for the stripped sync (fill) characters. When strip-sync bit is not selected (LOW), the sync character is assumed to be normal data and it is transferred into FIFO after parity checking. (When non-parity format is selected, parity is not checked.)

Strip Sync (C1 Bit 2)	WS0-WS2 (Data Format) (C2 Bit 3-5)	
1	X	No transfer of sync code. No parity check of sync code.
0	With Parity	*Transfer data and sync codes. Parity check.
0	Without Parity	*Transfer data and sync codes. No parity check.

*Subsequent to synchronization.

It is necessary to pay attention to the selected sync character in the following cases:

1. Data format is (6 + parity), (7 + parity).
2. Strip sync is not selected (LOW).
3. After synchronization when sync code is used as a fill character.

The transmitter sends the sync character without parity, but the receiver checks the parity as if it is normal data. Therefore, the sync character should be chosen to match the parity check selected for the receiver in this special case.

Receive Data First-In First-Out Register (Rx Data FIFO)

The receive data FIFO register consists of three 8-bit registers that are used for buffer storage of received data. Each 8-bit register has an internal status bit that monitors its full or empty condition. Data is always transferred from a full register to an adjacent empty register. The transfer from register to register occurs on E pulses. The RDA status bit will be HIGH when data is available in the last location of the Rx Data FIFO.

In an overrun condition, the overrunning character will be transferred into the full first stage of the FIFO register and will cause the loss of that data character. Successive overruns continue to overwrite the first register of the FIFO. This destruction of data is indicated by means of the overrun status bit. The overrun bit will be set when the overrun occurs and remains set until the status register is read, followed by a read of the Rx Data FIFO.

Unused data bits for short word lengths (including the parity bit) will appear as "0s" on the data bus when Rx Data FIFO is read.

Transmit Data First-In First-Out Register (Tx Data FIFO)

The transmit data FIFO register consists of three 8-bit registers that are used for buffer storage of data to be transmitted. Each 8-bit register has an internal status bit that monitors its full or empty condition. Data is always transferred from a full register to an adjacent empty register. The transfer is clocked by E pulses.

The TDRA status bit will be HIGH if the Tx Data FIFO is available for data.

Unused data bits for short word lengths will be handled as "don't cares". The parity bit is not transferred over the data bus, since the SSDA generates parity at transmission.

When an underflow occurs, the underflow character will be either the contents of the sync code register or an all-"1s" character. The underflow will be stored in the status register until cleared and will appear on the underflow output as a pulse approximately a Tx CLK HIGH period wide.

Status Register

The status register is an 8-bit, read-only register that provides the real-time status of the SSDA and the associated serial data channel. Reading the status register is a non-destructive process. The method of clearing status bits depends upon the function each bit represents and is discussed for each bit in the register.

Receiver Data Available (RDA), S Bit 0
The receiver data available status bit indicates when receiver data can be read from the Rx Data FIFO. The receiver data being present in the last register (No. 3) of the FIFO causes RDA to be HIGH for the 1-byte transfer mode. The RDA bit being HIGH indicates that the last two registers (No. 2 and No. 3) are full when in the 2-byte transfer mode. The second character can be read without a second status read (to determine that the character is available). An E pulse must occur between reads of the Rx Data FIFO to allow the FIFO to shift. Status must be read on a word-by-word basis if receiver data error checking is important. The RDA status bit is reset automatically when data is not available.

Transmitter Data Register Available (TDRA), S Bit 1
The TDRA status bit indicates that data can be loaded into the Tx Data FIFO register. The first register (No. 1) of the Tx Data FIFO being empty will be indicated by a HIGH level of the TDRA status bit in the 1-byte transfer mode. The first two registers (No. 1 and No. 2) must be empty for TDRA to be HIGH when in the 2-byte transfer mode. The Tx Data FIFO can be loaded with two bytes without an intervening status read; however, one E pulse must occur between loads. TDRA is inhibited by the Tx Reset or $\overline{\text{RESET}}$. When Tx Reset is set, the Tx Data FIFO is cleared and then released on the next E clock pulse. The Tx Data FIFO can then be loaded with up to three characters of data, even though TDRA is inhibited. This feature allows preloading data prior to the release of Tx Reset. A HIGH level on the $\overline{\text{CTS}}$ input inhibits the TDRA status bit in either sync mode of operation (one-sync-character or two-sync-character). $\overline{\text{CTS}}$ does not affect TDRA in the external sync mode. This enables the SSDA to operate under the control of the $\overline{\text{CTS}}$ input, with TDRA indicating the status of the Tx Data FIFO. The $\overline{\text{CTS}}$ input does not clear the Tx Data FIFO in any operating mode.

Data Carrier Detect ($\overline{\text{DCD}}$), S Bit 2
A positive transition on the $\overline{\text{DCD}}$ input is stored in the SSDA until cleared by reading both status and Rx Data FIFO. A "1" written into Rx Rs also clears the stored $\overline{\text{DCD}}$ status. The $\overline{\text{DCD}}$ status bit, when set, indicates that the $\overline{\text{DCD}}$ input has gone HIGH. The reading of both status and receive data FIFO allows bit 2 of subsequent status reads to indicate the state of the $\overline{\text{DCD}}$ input until the next positive transition.

Clear-to-Send ($\overline{\text{CTS}}$), S Bit 3
A positive transition on the $\overline{\text{CTS}}$ input is stored in the SSDA until cleared by writing a "1" into the Clear $\overline{\text{CTS}}$ control bit or the Tx Rs bit. The $\overline{\text{CTS}}$ status bit, when set, indicates that the $\overline{\text{CTS}}$ input has gone HIGH. The Clear $\overline{\text{CTS}}$ command (a "1" into C3 bit 2) allows bit 3 of subsequent status reads to indicate the state of the $\overline{\text{CTS}}$ input until the next positive transition.

Transmitter Underflow (TUF), S Bit 4
When data is not available for the transmitter, an underflow occurs and is so indicated in the status register (in the Tx Sync on underflow mode). The underflow status bit is cleared by writing a "1" into the clear underflow (CTUF) control bit or the Tx Rs bit. TUF indicates that a sync character will be transmitted as the next character. A TUF is indicated on the output *only* when the contents of the sync code register are to be transferred (transmit sync code on underflow = "1").

Receiver Overrun (Rx Ovrn), S Bit 5
Overrun indicates data has been received when the Rx Data FIFO is full, resulting in data loss. The Rx Ovrn status bit is set when overrun occurs. The Rx Ovrn status bit is cleared by reading status followed by reading the Rx Data FIFO or by setting the Rx Rs control bit.

Receiver Parity Error (PE), S Bit 6
The parity error status bit indicates that parity for the character in the last register of the Rx Data FIFO did not agree with selected parity. The parity error is cleared when the character to which it pertains is read from the Rx Data FIFO or when Rx Rs occurs. The $\overline{DCD}$ input does not clear the parity error or Rx Data FIFO status bits.

Interrupt Request ($\overline{IRQ}$), S Bit 7
The interrupt request status bit indicates when the $\overline{IRQ}$ output is in the active state ($\overline{IRQ}$ output = "0"). The IRQ status bit is subject to the same interrupt enables (RIE, TIE, and EIE) as the $\overline{IRQ}$ output. The IRQ status bit simplifies status inquiries for polling systems by providing single-bit indication of service requests.

Table 1 SSDA Programming Model

Register	Control Inputs		Address Control		Register Content							
	RS	R/W	AC2	AC1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Status (S)	0	1	X	X	Interrupt Request (IRQ)	Receiver Parity Error (PE)	Receiver Overrun (Rx Ovrn)	Transmitter Underflow (TUF)	Clear-to-Send ($\overline{CTS}$)	Data Carrier Detect ($\overline{DCD}$)	Transmitter Data Register Available (TDRA)	Receiver Data Available (RDA)
Control 1 C1	0	0	X	X	Address Control 2 (AC2)	Address Control 1 (AC1)	Receiver Interrupt Enable (RIE)	Transmitter Interrupt Enable (TIE)	Clear Sync	Strip Sync Characters (Strip Sync)	Transmitter Reset (Tx Rs)	Receiver Reset (Rx Rs)
Receive Data FIFO	1	1	X	X	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Control 2 (C2)	1	0	0	0	Error Interrupt Enable (EIE)	Transmit Sync Code on Underflow (Tx Sync)	Word Length Select 3 (WS3)	Word Length Select 2 (WS2)	Word Length Select 1 (WS1)	1-Byte/ 2-Byte Transfer (1-Byte/ 2-Byte)	Peripheral Control 2 (PC2)	Peripheral Control 1 (PC1)
Control 3 (C3)	1	0	0	1	Not Used	Not Used	Not Used	Not Used	Clear Transmitter Underflow Status (CTUF)	Clear $\overline{CTS}$ Status (Clear $\overline{CTS}$)	One-Sync-Character/ Two-Sync-Character Mode Control (1-Sync/ 2-Sync)	External/ Internal Sync Mode Control (E/I Sync
Sync Code	1	0	1	0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Transmit Data FIFO	1	0	1	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

X = Don't Care

Status Register

IRQ	Bit 7		The IRQ flag is cleared when the source of the IRQ is cleared. The source is determined by the enables in the control registers: TIE, RIE, EIE.
	Bits 6-0		Indicate the SSDA status at a point in time, and can be *reset* as follows:
	PE	Bit 6	Read Rx Data FIFO, or a "1" into Rx Rs (C1 bit 0).
	Rx Ovrn	Bit 5	Read status and then Rx Data FIFO, or a "1" into Rx Rs (C1 bit 0).
	TUF	Bit 4	A "1" into CTUF (C3 bit 3) or into Tx Rs (C1 bit 1)
	$\overline{CTS}$	Bit 3	A "1" into $\overline{CTS}$ (C3 bit 2) or a "1" into Tx Rs (C1 bit 1).
	$\overline{DCD}$	Bit 2	Read status and then Rx Data FIFO or a "1" into Rx Rs (C1 bit 0).
	TDRA	Bit 1	Write into Tx Data FIFO.
	RDA	Bit 0	Read Rx Data FIFO.

Control Register 1

AC2, AC1	Bits 7, 6	Used to access other registers, as shown above.
RIE	Bit 5	When "1", enables interrupt on RDA (S bit 0).
TIE	Bit 4	When "1", enables interrupt on TDRA (S bit 1).
Clear Sync	Bit 3	When "1", clears receiver character synchronization
Strip Sync	Bit 2	When "1", strips all sync codes from the received data stream.
Tx Rs	Bit 1	When "1", resets and inhibits the transmitter section.
Rx Rs	Bit 0	When "1", resets and inhibits the receiver section.

Control Register 2

EIE	Bit 7	When "1", enables the PE, Rx Ovrn, TUF, $\overline{CTS}$, and $\overline{DCD}$ interrupt flags (S bits 6 through 2).
Tx Sync	Bit 6	When "1", allows sync code content to be transferred on underflow, and enables the TUF status bit and output. When "0", an all-mark character is transmitted on underflow.
WS3, 2, 1	Bits 5-3	Word Length Select

Bit 5 WS3	Bit 4 WS2	Bit 3 WS1	Word Length
0	0	0	6 Bits + Even Parity
0	0	1	6 Bits + Odd Parity
0	1	0	7 Bits
0	1	1	8 Bits
1	0	0	7 Bits + Even Parity
1	0	1	7 Bits + Odd Parity
1	1	0	8 Bits + Even Parity
1	1	1	8 Bits + Odd Parity

1-Byte/2-Byte	Bit 2	When "1", enables the TDRA and RDA bits to indicate when a 1-byte transfer can occur; when "0", the TDRA and RDA bits indicate when a 2-byte transfer can occur.
PC2, PC1	Bits 1-0	SM/$\overline{DTR}$ Output Control

Bit 1 PC2	Bit 0 PC1	SM/$\overline{DTR}$ Output at Pin 5
0	0	1
0	1	Pulse ⎍, 1-Bit Wide, on SM
1	0	0
1	1	SM Inhibited, 0

Control Register 3

CTUF	Bit 3	When "1", clears TUF (S bit 4) and IRQ, if enabled.
Clear $\overline{CTS}$	Bit 2	When "1", clears $\overline{CTS}$ (S bit 3) and IRQ, if enabled.
1-Sync/2-Sync	Bit 1	When "1", selects the one-sync-character mode; when "0", selects the two-sync-character mode.
E/I Sync	Bit 0	When "1", selects the external sync mode; when "0", selects the internal sync mode.

Notes
When the SSDA is used in applications requiring the MSB of data to be received and transmitted first, the data bus inputs to the SSDA may be reversed (D_0 to D_7, etc.). Caution must be used when this is done, since the bit positions in this table will be reversed, and the parity should not be selected.

Absolute Maximum Ratings

Supply Voltage	-0.3 V, +7.0 V
Input Voltage	-0.3 V, +7.0 V
Operating Temperature Range	
F6852P, S, F68A52P, S, F68B52P, S	0°C, +70°C
F6852CP, CS, F68A52CP, CS	-40°C, +85°C
F6852DLQB	-55°C, +85°C
F6852DMQB	-55°C, +125°C
Storage Temperature Range	-55°C, +150°C
Thermal Resistance	
Plastic Package	120°C/W
Ceramic Package	60°C/W

Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = 0 to 70°C unless otherwise noted

Symbol	Characteristic	Signal	Min	Typ	Max	Unit	Test Conditions
V_{IH}	Input HIGH Voltage		2.0			V	
V_{IL}	Input LOW Voltage				0.8	V	
I_{IN}	Input Leakage Current	Tx CLK, Rx CLK, Rx DATA, Enable, $\overline{RESET}$, RS, R/$\overline{W}$, $\overline{CS}$, $\overline{DCD}$, $\overline{CTS}$		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	3-State (OFF State) Input Current	D_0-D_7		2.0	10	μA	V_{IN} = 0.4 to 2.4 V, V_{CC} = 5.25 V
V_{OH}	Output HIGH Voltage	D_0-D_7 Tx DATA, $\overline{DTR}$, TUF	2.4 2.4			V	I_{Load} = -205 μA, Enable Pulse Width <25 μs I_{Load} = -100 μA, Enable Pulse Width <25 μs
V_{OL}	Output LOW Voltage				0.4	V	I_{Load} = 1.6 mA, Enable Pulse Width <25 μs
I_{LOH}	Output Leakage Current (OFF State)	($\overline{IRQ}$)		1.0	10	μA	V_{OH} = 2.4 V
P_D	Power Dissipation			300	525	mW	
C_{IN}	Input Capacitance	 D_0-D_7 All Other Inputs			 12.5 7.5	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance	Tx DATA, SM/$\overline{DTR}$, TUF $\overline{IRQ}$			10 5.0	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz

AC Characteristics V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = 0°C to +70°C unless otherwise noted.

Symbol	Characteristic	F6852		F68A52		F68B52		Unit	Test Condition
		Min	Max	Min	Max	Min	Max		
PW_{CL}	Minimum Clock Pulse Width, LOW *(Figure 1)*	700		400		280		ns	
PW_{CH}	Minimum Clock Pulse Width, HIGH *(Figure 2)*	700		400		280		ns	
f_C	Clock Frequency		600		1000		1500	kHz	
t_{RDSU}	Receive Data Set-up Time *(Figures 3, 7)*	350		200		160		ns	
t_{RDH}	Receive Data Hold Time *(Figure 3)*	350		200		160		ns	
t_{SM}	Sync Match Delay Time *(Figure 3)*		1.0		0.666		0.500	μs	
t_{TDD}	Clock-to-Data Delay for Transmitter *(Figure 4)*		1.0		0.666		0.500	μs	
t_{TUF}	Transmitter Underflow *(Figure 4, 6)*		1.0		0.666		0.500	μs	
t_{DTR}	$\overline{DTR}$ Delay Time *(Figure 5)*		1.0		0.666		0.500	μs	
t_{IR}	Interrupt Request Release Time *(Figure 5)*		1.2		0.800		0.600	μs	
t_{Res}	$\overline{RESET}$ Minimum Pulse Width	1.0		0.666		0.500		μs	
t_{CTS}	$\overline{CTS}$ Set-up Time *(Figure 6)*	200		150		120		ns	
t_{DCD}	$\overline{DCD}$ Set-up Time *(Figure 7)*	500		350		250		ns	
t_r, t_f	Input Rise and Fall Times (except Enable)		1.0*		1.0*		1.0*	μs	0.8 V to 2.0 V

*1.0 μs or 10% of the pulse width, whichever is smaller.

5

Bus Timing Characteristics

Read *(Figures 8 and 10)*

Symbol	Characteristic	F6852		F68A52		F68B52		Unit
		Min	Max	Min	Max	Min	Max	
t_{cycE}	Enable Cycle Time	1.0		0.666		0.5		μs
PW_{EH}	Enable Pulse Width, HIGH	0.45	25	0.28	25	0.22	25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.28		0.21		μs
t_{AS}	Set-up Time, Address and R/$\overline{W}$ valid to Enable positive transition	160		140		70		ns
t_{DDR}	Data Delay Time		320		220		180	ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable input		25		25		25	ns

Write *(Figures 9 and 10)*

t_{cycE}	Enable Cycle Time	1.0		0.666		0.5		μs
PW_{EH}	Enable Pulse Width, HIGH	0.45	25	0.28	25	0.22	25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.28		0.21		μs
t_{AS}	Set-up Time, Address and R/$\overline{W}$ valid to Enable positive transition	160		140		70		ns
t_{DSW}	Data Set-up Time	195		80		60		ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable input		25		25		25	ns

Fig. 1 Clock Pulse Width, Low-State

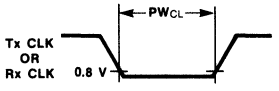


Fig. 2 Clock Pulse Width, High-State

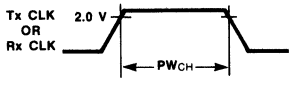
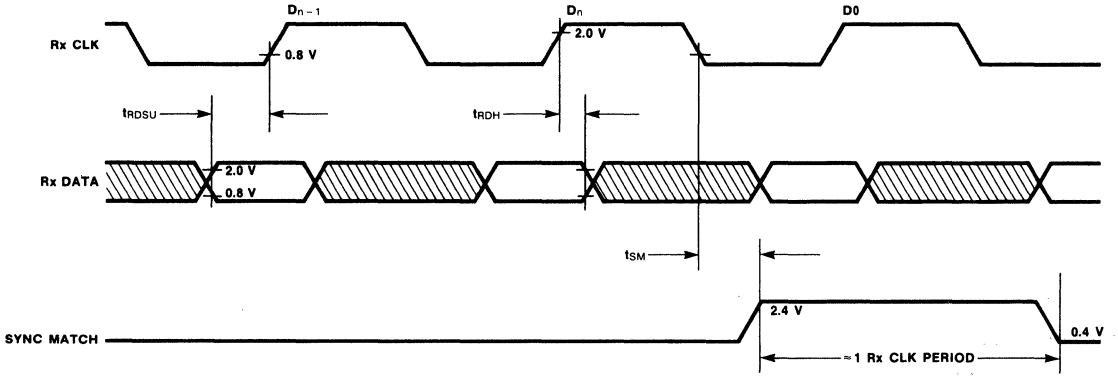


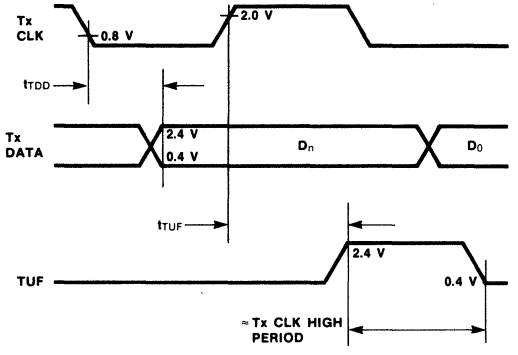
Fig. 3 Receive Data Set-up and Hold Times and Sync Match Delay Time



n = Number of bits in character

= Don't care

Fig. 4 Transmit Data Output Delay and Transmitter Underflow Delay Time



n = Number of bits in character

Fig. 5 Data Terminal Ready and Interrupt Request Release Times

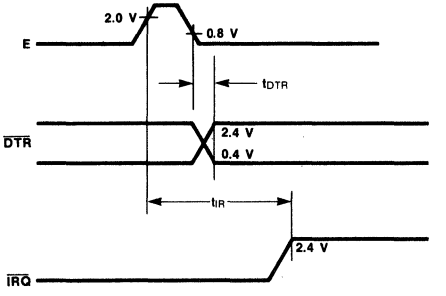


Fig. 6 Clear-to-Send Set-up Time

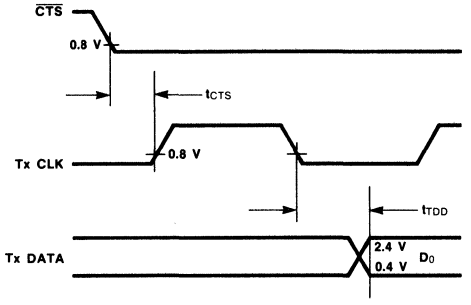


Fig. 7 Data Carrier Detect Set-up Time

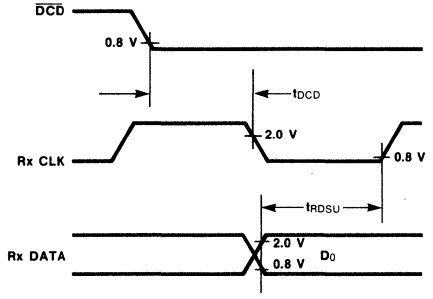


Fig. 8 Bus Read Timing Characteristics (Read information from SSDA)

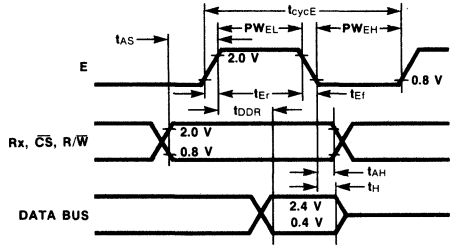


Fig. 9 Bus Write Timing Characteristics (Write information into SSDA)

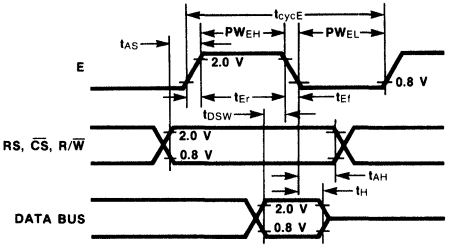
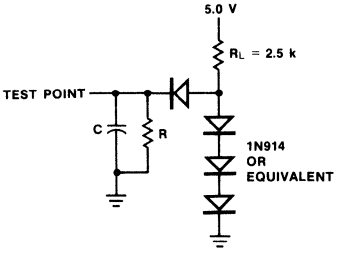


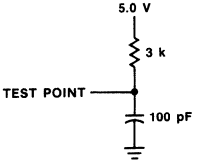
Fig. 10 Bus Timing Test Loads

Load A
(D_0-D_7, $\overline{DTR}$, Tx DATA, TUF)



C = 130 pF for D_0-D_7
= 30 pF for $\overline{DTR}$, Tx DATA, and TUF
R = 11.7 kΩ for D_0-D_7
= 24 kΩ for $\overline{DTR}$, Tx DATA, and TUF

Load B
($\overline{IRQ}$ Only)



Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6852 P,S	0°C to 70°C
	F6852 CP,CS	-40°C to +85°C
	F6852 DLQB	-55°C to +85°C
	F6852 DMQB	-55°C to +125°C
1.5 MHz	F68A52 P,S	0°C to 70°C
	F68A52 CP,CS	-40°C to +85°C
2.0 MHz	F68B52 P,S	0°C to 70°C

P = Plastic package; S = Ceramic package

FAIRCHILD
A Schlumberger Company

F6854/F68A54/F68B54 Advanced Data Link Controller (ADLC)

Microprocessor Product

Description

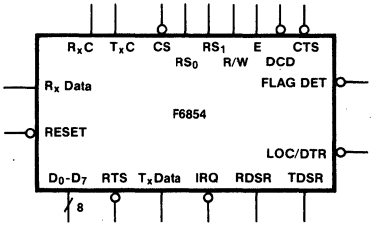
The F6854/F68A54/F68B54 Advanced Data Link Controllers (ADLC) perform the complex MPU/data communication link function for the Advanced Data Communication Control Procedure (ADCCP), High-level Data Link Control (HDLC) and Synchronous Data Link Control (SDLC) standards. The ADLC provides key interface requirements with improved software efficiency. The ADLC is designed to provide the data communications interface for both primary and secondary stations in stand-alone, polling and loop configurations.

- **F6800 Compatible**
- **Protocol Features**
  - **Automatic Flag Detection and Synchronization**
  - **Zero Insertion and Deletion**
  - **Automatic Address Field Extension (Optional)**
  - **Extended Control Field (Optional)**
  - **Auto Extendable Logic Control Field (Optional)**
  - **Variable Word Length Information Field 5-, 6-, 7-, or 8-Bit**
  - **Automatic Frame Check Sequence Generation and Check**
  - **Abort Detection and Transmission**
  - **Idle Detection and Transmission**
- **Modem/Data Channel Control Lines**
- **Loop Mode**
- **Single 5 V Power Supply**
- **Enhanced Speed Options**
  F6854—1.0 MHz
  F68A54—1.5 MHz
  F68B54—2.0 MHz

Pin Names

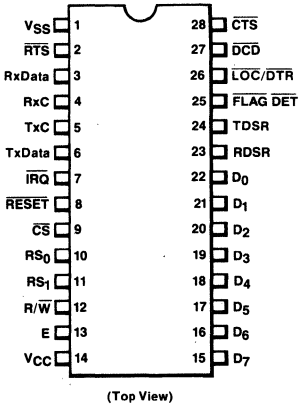
RxData	Receiver Serial Data Input
RxC	Receiver Data Timing Clock Input
TxC	Transmitter Data Timing Clock Input
$\overline{\text{RESET}}$	Chip Master Reset Input
$\overline{\text{CS}}$	Chip Select Input
RS_0, RS_1	Register Addressing Select Inputs
R/$\overline{\text{W}}$	Read/Write Input
E	System Control Clock Input
$\overline{\text{DCD}}$	Data Carrier Detect Input
$\overline{\text{CTS}}$	Clear-to-Send Input
D_0 - D_7	Bidirectional Data I/O Lines
$\overline{\text{RTS}}$	Request-to-Send Output
TxData	Transmitter Serial Data Output
$\overline{\text{IRQ}}$	Interrupt Request Output
RDSR	Receiver Data Service Request Output
TDSR	Transmitter Data Service Request Output
$\overline{\text{FLAG}}$ $\overline{\text{DET}}$	Flag Detect Output
$\overline{\text{LOC}}$/$\overline{\text{DTR}}$	Loop On-line Control/Data Terminal Ready Output
V_{SS}	Ground
V_{CC}	+5 V Power Supply

Logic Symbol



V_{CC} = Pin 14
V_{SS} = Pin 1

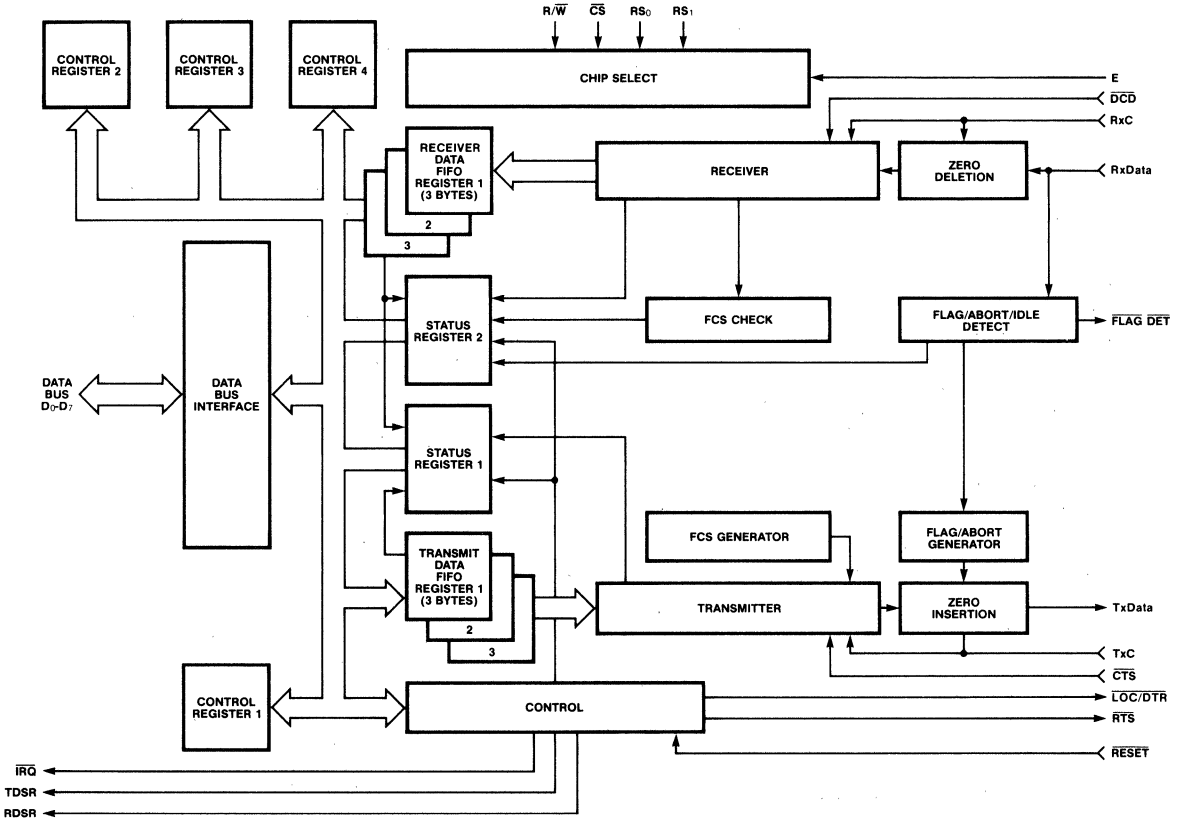
Connection Diagram
28-Pin DIP



(Top View)

5

Block Diagram



Operation

Initialization

During a power-on sequence, the ADLC is reset via the Reset ($\overline{\text{RESET}}$) input and internally latched in a reset condition to prevent erroneous output transitions. The four Control Registers must be programmed prior to the release of the reset condition. The release of the reset condition is performed via software by writing a LOW into the Receiver Reset (RxRS) control bit and/or Transmitter Reset (TxRS) control bit. The release of the reset condition will be done after $\overline{\text{RESET}}$ has gone HIGH.

At any time during operation, writing a HIGH into the RxRS control bit or TxRS control bit causes the reset condition of the receiver or transmitter section.

Transmitter Operation

The Transmitter Data FIFO Register (Tx FIFO) cannot be pre-loaded when the transmitter section is in a reset state. After the reset release, the Flag/Mark Idle Select control bit (F/M Idle) selects either the mark idle state (inactive idle) or the flag time fill (active idle) state. This active or inactive mark idle state will continue until data is loaded into the Tx FIFO.

The availability of the Tx FIFO is indicated by the Transmitter Data Available/Frame Complete (TDRA/FC) status bit under the control of the 2-Byte/1-Byte Transfer (2/1-Byte) control bit. TDRA status is inhibited by the TxRS control bit or Clear-to-Send ($\overline{\text{CTS}}$) input being HIGH. When the 1-byte mode is selected, one byte of the Tx FIFO is available for data transfer when TDRA/FC goes

HIGH. When the 2-byte mode is selected, two successive bytes can be transferred when TDRA/FC goes HIGH.

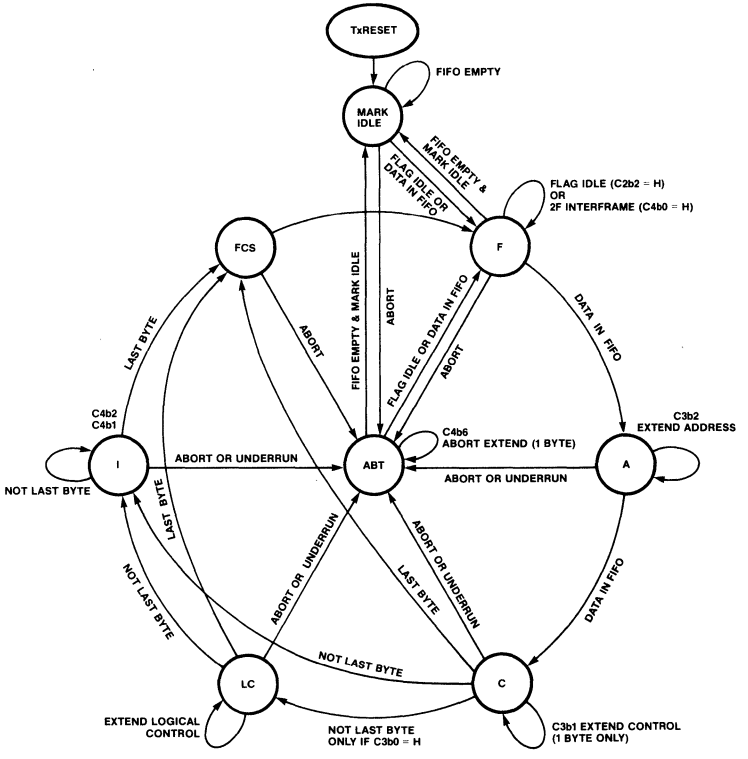
The first byte (address field) should be written into the Tx FIFO at the frame-continue address. Then the transmission of a frame automatically starts. If the Transmitter is in a mark idle state, the transfer of an address causes an opening flag within two or three transmitter clock cycles. If the Transmitter has been in a time fill state, the current time fill flag being transmitted is assumed as an opening flag and the address field will follow it.

A frame continues as long as data is written into the Tx FIFO at the frame-continue address. The ADLC internally keeps track of the field sequence in the frame. The frame format is described in the Frame Format section.

The frame is terminated by one of two methods. The most efficient way to terminate the frame from a software standpoint is to write the last data character into the Tx FIFO frame-terminate address (RS_1, RS_0 = HH) rather than the Tx FIFO frame-continue address (RS_1, RS_0 = HL). An alternate method is to follow the last write of data in the Tx FIFO frame-continue address with the setting of the Transmit Last Data (Tx Last) control bit. Either method causes the last character to be transmitted and the Frame Check Sequence (FCS) field to be appended automatically along with a closing flag. Data for a new frame can be loaded into the Tx FIFO immediately after the old frame data if TDRA/FC is HIGH. The closing flag can serve as the opening flag of the next frame, or separate opening and closing flags may be transmitted. If a new frame is not ready to be transmitted, the ADLC will automatically transmit the active (flag) or inactive (mark) idle condition.

5

ADLC Transmitter State Diagram
(Cibi refers to Control Register bit)



Data Being Transmitted:
F = flag
A = address
C = (link) control

LC = logical control (optional)
I = information
FCS = frame check sequence
ABT = abort

If the Tx FIFO becomes empty at any time during frame transmission (the Tx FIFO has no data to transfer into the transmitter shift register during transmission of the last half of the next-to-last bit of a word), an underrun will occur and the Transmitter automatically terminates the frame by transmitting an abort. The underrun state is indicated by the Transmitter Underrun (TxU) status bit.

Any time the Transmit Abort (ABT) control bit is set, the Transmitter immediately aborts the frame (transmits at least eight consecutive 1s) and clears the Tx FIFO. If the Abort Extend (ABT_{EX}) control bit is set at the time, an idle (at least 16 consecutive 1s) is transmitted. An abort or idle in an out-of-frame condition can be useful to gain eight or 16 bits of delay. (For an example see Programming Considerations.)

The $\overline{CTS}$ input and Request-to-Send ($\overline{RTS}$) output are provided for a modem or other hardware interface.

The TDRA/FC status bit (when selected to be frame-complete status) can cause an interrupt upon frame completion (i.e., a flag or abort completion).

Details regarding the pin functions, Tx FIFO operation, and control and status registers are described in their respective sections.

Receiver Operation

Data and a pre-synchronized clock are provided to the ADLC receiver section by means of the Receiver Serial Data (RxData) and Receiver Data Timing Clock (RxC) inputs. The data is a continuous stream of binary bits with the characteristic that a maximum of five 1s can occur in succession unless abort, flag, or idling conditions occur. The Receiver continuously (on a bit-by-bit basis) searches for flags and aborts.

When a flag is detected, the Receiver establishes frame synchronization to the flag timing. If a series of flags is received, the Receiver resynchronizes to each flag.

If the frame is terminated before the internal buffer time expires (the frame data is less than 25 bits after an opening flag), the frame is simply ignored. Noise on RxData during time fill can cause this kind of invalid frame.

Once synchronization has been achieved and the internal buffer time (24 bit-times) expires, data will automatically transfer to the Receiver Data FIFO Register (RxFIFO). The Rx FIFO is clocked by System Control Clock (E) input to cause received data to move through the Rx FIFO to the last empty register location. The Receiver Data Available (RDA) status bit indicates when data is present in the last register (Register 3) for the 1-byte transfer mode. The 2-byte transfer mode causes the RDA status bit to indicate data is available when the last two Rx FIFO register locations (Registers 2 and 3) are full. If the data character present in the Rx FIFO is an address octet, Status Register 1 will exhibit an address present status condition. Data being available in the Rx FIFO causes an interrupt to be initiated (assuming the Receiver Interrupt Enable (RIE) control bit is enabled, RIE="1"). The MPU will read the ADLC status registers as a result of the interrupt or in its turn in a polling sequence. The Receiver Data Available (RDA) or Address Present (AP) status bits will indicate that receiver data is available and the MPU should subsequently read the Rx FIFO. The Interrupt Request (IRQ) and RDA status bits will then be reset automatically. If more than one character is received and is resident in the Rx FIFO, subsequent E clocks will cause the Rx FIFO to update and the RDA and IRQ status bits will again be set. In the 2-byte transfer mode both data bytes may be read on consecutive E cycles. The AP status bit provides for 1-byte transfers only.

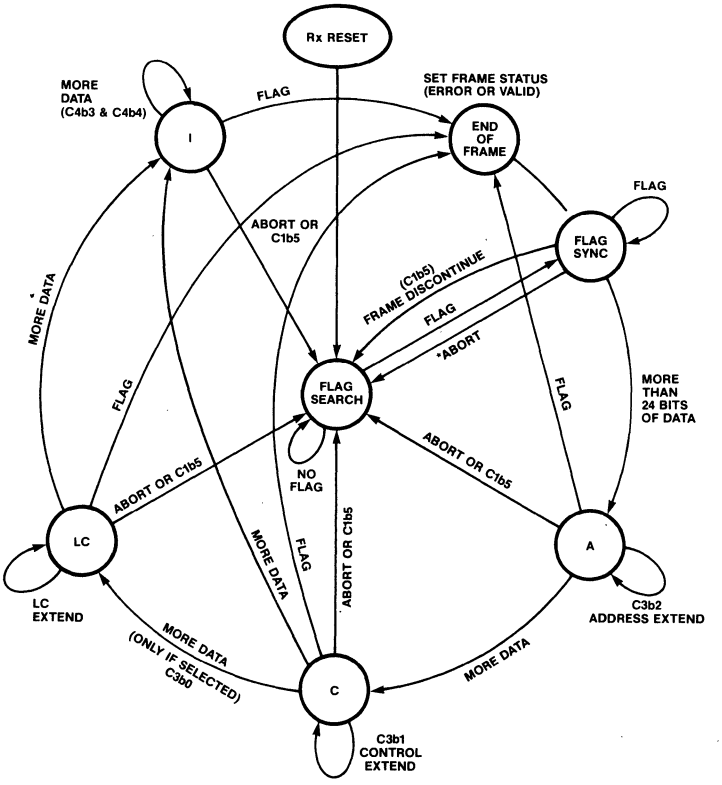
The sequence of each field in the received frame is automatically handled by the ADLC. The frame format is described in the Frame Format section.

When a closing flag is received, the frame is terminated. The 16 bits preceding the closing flag are regarded as the FCS and are not transferred to the MPU. Whatever data is present in the most significant byte portion of the receiver buffer register is right justified and transferred to the Rx FIFO. The frame boundary pointer, explained in the Rx FIFO Register section, is set simultaneously in the Rx FIFO. The frame boundary pointer sets the Frame Valid (FV) status bit (when the frame was completed with no error) or the Frame Check Sequence/Invalid Frame Error (ERR) status bit (when the frame was completed with error) when the last byte of the frame appears at the last location of the Rx FIFO. As long as the FV or ERR status bit is set, the data transfer from the second location of the Rx FIFO to the last location of the Rx FIFO is inhibited.

Any time the Rx Frame Discontinue (DISCONTINUE) control bit is set, the ADLC discards the current frame data in the ADLC without dropping flag synchronization. This feature can be used to ignore a frame which is addressed to another station.

The reception of an abort or idle is explained in the Frame Format section. The details regarding the pin functions, Rx FIFO operation, and control and status registers are described in their respective sections.

ADLC Receiver State Diagram

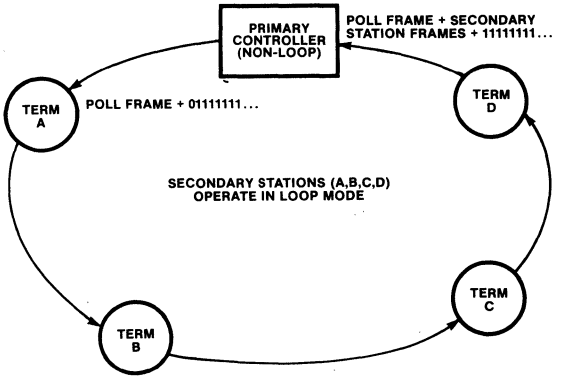


*Out-of-frame Abort (No IRQ)

Loop Mode Operation

In the loop mode the ADLC not only transmits and receives data frames in the manner previously described, but also has additional features for gaining and relinquishing loop control. In *Figure 1*, a configuration is shown which depicts loop mode operation. The system configuration shows a primary station and several secondary stations. The loop is always under control of the primary station. When the primary wants to receive data, it transmits a poll sequence and allows frame transmission to secondary stations on the loop. Each secondary is in series and adds one bit of delay to the loop. Secondary A in the figure receives data from the primary via the RxData input, delays the data one bit, and transmits it to secondary B via the Transmitter Serial Data (TxData) output. Secondaries B, C, and D operate in a similar manner. Therefore, data passes through each secondary and is received back by the primary controller.

Fig. 1 Typical Loop Configuration



Certain protocol rules must be followed that establish the manner by which the secondary station places itself on-loop (connects TxData to the loop), goes active on the loop (starts transmitting data on the loop), and goes off the loop (disconnects TxData). Otherwise, loop data to other stations down-loop would be interrupted. The data stream always flows the same way; the order in which secondary terminals are serviced is determined by the hardware configuration. The primary controller times the delay through the loop. Should it exceed n + 1 bit-times, where n is the number of secondary terminals on the loop, it will indicate a loop failure. Control is transferred to a secondary by transmitting a go-ahead signal following the closing flag of a polling frame (request for a response from the secondary) from the primary station. The go-ahead from the primary is a 0 and seven 1s followed by mark idling. The primary can abort its response request by interrupting its idle with flags. The secondary should immediately stop transmission and return control to the primary. When the secondary completes its frame, a closing flag is transmitted followed by all 1s. The primary detects the final 01111111 (go-ahead to the primary) and resumes control. Note that if a down-loop secondary (e.g., station D) needs to insert information following an up-loop station (e.g., station A), the go-ahead to station D is the last 0 of the closing flag from station A followed by 1s.

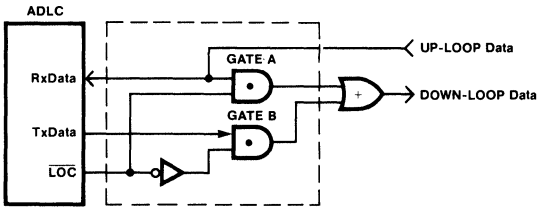
The ADLC in the primary station should operate in a non-loop, full-duplex mode. The ADLC in the secondaries should operate in a loop mode, monitoring up-loop data on RxData. The ADLC can recognize the necessary sequences in the data stream to automatically go on/off the loop and to insert its own station data. This procedure is summarized in *Table 1.*

1. Go On-loop — When the ADLC powers up, the terminal station will be off line. The first task is to become an active terminal on the loop. The ADLC must be connected to a loop link via an external switch as shown in *Figure 2.* After a hardware reset, the ADLC Loop On-line Control/Data Terminal Ready ($\overline{LOC}/\overline{DTR}$) output will be in the HIGH state and the up-loop receive data repeated through gate A to the down-loop stations. Any up-loop transmission will be received by the ADLC. The Loop/Non-Loop Mode (LOOP) control bit must be set to place the ADLC in the loop mode. The ADLC now monitors its RxData input for a string of seven consecutive 1s which will allow a station to go on line. The loop operation may be monitored by use of the Loop Status (LOOP) status bit. After power-up and reset, this bit is a LOW. When seven consecutive 1s are received by the ADLC, the $\overline{LOC}/\overline{DTR}$ output will go to a LOW level, disabling gate A (refer to *Figure 2),* enabling gate B and connecting the ADLC TxData output to the down-loop stations. The up-loop data is now repeated to the down-loop stations via the ADLC. A 1-bit delay is inserted in the data (in NRZI mode, there will be a 2-bit delay) as it circulates through the ADLC. The ADLC is now on-line and the LOOP status bit will be at a HIGH.

Table 1 Summary of Loop Mode Operation

State	Receiver (Rx) Section	Transmitter (Tx) Section	LOOP Status Bit
Off-loop	Rx section receives data from loop and searches for seven 1s (when the LOC/DTR control bit set) to go on-loop.	Inactive 1. NRZ Mode TxData output is maintained HIGH (mark). 2. NRZI Mode TxData output reflects the RxData input state delayed by one bit-time. (Not normally connected to loop.) The NRZI data is internally decoded to provide error-free transitions to on-loop mode.	L
On-loop	1. When GAP/TST control bit is set, Rx section searches for 01111111 pattern (the EOP or go-ahead) to become the active terminal on the loop. 2. When the LOC/DTR control bit is reset, Rx section searches for eight 1s to go off-loop.	Inactive 1. NRZ Mode TxData output reflects RxData input state delayed one bit-time. 2. NRZI Mode TxData output reflects RxData input state delayed two bit-times.	H
Active	Rx section searches for flag (an interrupt from the loop controller) at RxData input. Received flag causes FLAG DET output to go LOW. IRQ is generated if the RIE and FDSE control bits are set.	TxData originates within ADLC until GAP/TST control bit is reset and a flag or abort is completed, then returns to on-loop state.	L

Fig. 2 External Loop Logic



2. Go Active after Poll — The receiver section will monitor the up-link data for a general or addressed poll command; the Tx FIFO should be loaded with data so that when the go-ahead sequence of a 0 followed by seven 1s (01111111--) is detected, transmission can be initiated immediately. When the polling frame is detected, the Go Active On Poll/Test (GAP/TST) control bit must be set. A minimum of seven bit-times are available to set this control bit after the closing flag of the poll. When the go-ahead is detected by the Receiver, the ADLC will automatically change the seventh 1 to a 0 so that the repeated sequence out gate B in *Figure 2* is now an opening flag sequence (01111110). Transmission now continues from the Tx FIFO with data (address, control, etc.) as previously described. When the ADLC has gone active-on-poll, the LOOP status bit will go to a LOW. The Receiver searches for a flag, which indicates that the primary station is interrupting the current operation.

3. Go Inactive when On-loop — The GAP/TST control bit may be reset at any time during transmission. When the frame is complete (the closing flag or abort is transmitted), the loop is automatically released and the station reverts back to being just a 1-bit delay in the loop, repeating up-link data. If the GAP/TST control bit is not reset by software and the final frame is transmitted (F/M Idle control bit = LOW), then the Transmitter will mark idle and will not release the loop to up-loop data. A transmitter abort command would have to be used in this case in order to go inactive when on the loop. Also, if the Tx FIFO was not preloaded with data (address, control, etc.) prior to changing the go-ahead character to a flag, the ADLC will either transmit flags (active idle character) until data is loaded (when the F/M Idle control bit is HIGH) or will go into an underrun condition and transmit an abort (when the F/M Idle control bit is LOW). When an abort is transmitted, the GAP/TST control bit is reset automatically and the ADLC reverts to its repeating mode (TxData = delayed RxData). When the ADLC Transmitter lets go of the loop, the LOOP status bit will return to a HIGH, indicating normal on-loop retransmission of up-loop data.

4. Go Off-loop — The ADLC can drop off the loop (go off-line) similar to the way it went on-line. When the Loop On-line Control/DTR Control (LOC/DTR) control bit is reset, the ADLC receiver section looks for eight successive "1s" before allowing the $\overline{LOC}/\overline{DTR}$ output to return HIGH (the inactive state). Gate A in *Figure 2* will be enabled and gate B disabled allowing the loop to maintain continuity without disturbance. The LOOP status bit will show an off-line condition (logic HIGH).

Pin Functions

All inputs of the ADLC are high-impedance and TTL-level compatible. All outputs of the ADLC are compatible with standard TTL. Interrupt Request ($\overline{IRQ}$), however, is an open-drain output (no internal pull-up).

5

Interface for MPU

Bidirectional Data I/O Lines (D_0-D_7)
These data bus I/O ports allow the data transfer between ADLC and system bus. The data bus drivers are 3-state devices that remain in the high-impedance (OFF) state except when the MPU performs an ADLC Read operation.

System Control Clock (E)
E activates the address inputs ($\overline{CS}$, RS_0 and RS_1) and Read/Write input ($R/\overline{W}$) and enables the data transfer on the data bus. E also moves data through the Tx FIFO and Rx FIFO. E should be a free-running clock, such as the F6800 MPU system clock.

Chip Select ($\overline{CS}$)
An ADLC Read or Write operation is enabled only when the $\overline{CS}$ input is LOW and the E input is HIGH ($E{\cdot}\overline{CS}$).

Register Addressing Select (RS_0, RS_1)
When the RS_0, RS_1 inputs are enabled by ($E{\cdot}\overline{CS}$), they select internal registers in conjunction with the $R/\overline{W}$ input and Address Control (AC) control bit. Register addressing is defined in *Table 2*.

Read/Write ($R/\overline{W}$)
The $R/\overline{W}$ input controls the direction of data flow on the data bus when it is enabled by ($E{\cdot}\overline{CS}$). The bidirectional Data Bus Interface acts as an output driver when $R/\overline{W}$ is HIGH, and as an input buffer when LOW. It also selects the read-only and write-only registers within the ADLC.

Chip Master Reset ($\overline{RESET}$)
The $\overline{RESET}$ input provides a means of resetting the ADLC from a hardware source. In the LOW state, the $\overline{RESET}$ input causes the following:

1. RxRS and TxRS are set, causing both the receiver and transmitter sections to be held in a reset condition.

2. Resets the following control bits: ABT, Request-to-Send Control (RTS), LOOP, and LOC/DTR.
3. Clears all stored status condition of the status registers.
4. The $\overline{RTS}$ and $\overline{LOC/DTR}$ ouputs go HIGH; TxData goes to the mark state (1s are transmitted).

When $\overline{RESET}$ returns HIGH (the inactive state), the transmitter and receiver sections will remain in the reset state until TxRS and RxRS are cleared via the data bus under software control. The control register bits affected by $\overline{RESET}$ cannot be changed when $\overline{RESET}$ is LOW.

Interrupt Request Output ($\overline{IRQ}$)
$\overline{IRQ}$ will be LOW if an interrupt situation exists and the appropriate interrupt enable has been set. The interrupt remains as long as the cause for the interrupt is present and E is set.

Clock and Data of Transmitter and Receiver
Transmitter Data Timing Clock (TxC)
The Transmitter shifts data on the negative transition of the TxC input. When the loop mode or test mode is selected, TxC should be the same frequency and phase as the RxC input. The data rate of the Transmitter should not exceed the E frequency.

Receiver Data Timing Clock (RxC)
The Receiver samples the data on the positive transition of the TxC input. RxC should be synchronized with RxData externally.

Transmitter Serial Data (TxData)
The serial data from the Transmitter is coded in NRZ or NRZI (zero complement) data format.

Receiver Serial Data (RxData)
The serial data to be received by the ADLC can be coded in NRZ or NRZI (zero complement) data format. The data rate of the Receiver should not exceed the E frequency. If a partial byte reception is possible at the end of a frame, the maximum data rate of the Receiver is indicated by the following relationship:

$$f_{RxC} \leq \frac{1}{2t_E + 300\ ns}$$

where t_E is the period of E.

Peripheral/Modem Control
Request-to-Send ($\overline{RTS}$)
The Request-to-Send output is controlled by the RTS control bit in conjunction with the state of the transmitter section. When the RTS control bit goes HIGH, the $\overline{RTS}$ output is forced LOW. When the RTS control bit returns LOW, the $\overline{RTS}$ output remains LOW until the end of the frame and there is no further data in the Tx FIFO for a new frame. The positive transition of $\overline{RTS}$ occurs after the completion of a flag, an abort, or when the RTS control bit is reset during a mark idling state. When the $\overline{RESET}$ input is LOW, the $\overline{RTS}$ output goes HIGH.

Clear-to-Send ($\overline{CTS}$)
The $\overline{CTS}$ input provides a real-time inhibit to the TDRA/FC status bit and its associated interrupt. The positive transition of $\overline{CTS}$ is stored within the ADLC to insure its occurrence will be acknowledged by the system. The stored CTS information and its associated IRQ status bit (if enabled) are cleared by writing a HIGH in the Clear Transmitter Status (CLR TxST) or the TxRS status bit.

Data Carrier Detect ($\overline{DCD}$)
The $\overline{DCD}$ input provides a real-time inhibit to the receiver section. A HIGH level on the $\overline{DCD}$ input resets and inhibits the receiver register, but data in the Rx FIFO from a previous frame is not disturbed. The positive transition of $\overline{DCD}$ is stored within the ADLC to insure that its occurrence will be acknowledged by the system. The stored DCD information and its associated IRQ status bit (if enabled) are cleared by means of the Clear Receiver Status (CLR RxST) or by the RxRS control bit.

Loop On-line Control/Data Terminal Ready ($\overline{LOC/DTR}$)
The $\overline{LOC/DTR}$ output serves as a $\overline{DTR}$ output in the non-loop mode or as an $\overline{LOC}$ output in the loop mode. When the $\overline{LOC/DTR}$ output performs the $\overline{DTR}$ function, it is turned on and off by means of the LOC/DTR control bit.

When the LOC/DTR control bit is HIGH, the $\overline{DTR}$ output will be LOW. In the loop mode the $\overline{LOC/DTR}$ output provides the means of controlling the external loop interface hardware to go on-line or off-line. When the LOC/DTR control bit is set and the loop has idled for seven bit-times or more (RxData = 01111111...), the $\overline{LOC/DTR}$ output will go LOW (on-line). When the LOC/DTR control bit is LOW and the loop has idled for eight bit-times or more, the $\overline{LOC/DTR}$ output will return HIGH (off-line). The $\overline{RESET}$ input being LOW will cause the $\overline{LOC/DTR}$ output to be HIGH.

Flag Detect ($\overline{FLAG\ DET}$)
The $\overline{FLAG\ DET}$ output indicates the reception of a flag and initiates an external time-out counter for the loop mode operation. The $\overline{FLAG\ DET}$ output goes LOW for one bit-time beginning at the last bit of the flag character, as sampled by RxC.

DMA Interface
Receiver Data Service Request (RDSR)
The RDSR output is provided primarily for use in DMA mode operation and indicates (when HIGH) that the Rx FIFO requests service (RSDR output reflects the RDA

status bit). If the prioritized status mode is selected, RDSR will be inhibited when any other receiver status conditions are present. RDSR goes LOW when the Rx FIFO is read.

Transmitter Data Service Request (TDSR)
The TDSR output is provided for DMA mode operation and indicates (when HIGH) that the Tx FIFO requests service. TDSR goes LOW when the Tx FIFO is loaded. TDSR is inhibited by the TxRS control bit being set, $\overline{RESET}$ being LOW, or $\overline{CTS}$ being HIGH. If the prioritized status mode is used, the TxU status bit also inhibits TDSR. TDSR reflects the TDRA/FC status bit except in the frame-complete mode.

ADLC Registers

Eight registers in the ADLC can be accessed via D_0-D_7 and RS_0, RS_1. The registers are defined as read-only or write-only according to the direction of information flow. The addresses of these registers are defined in *Table 2*. The Tx FIFO can be accessed by two different addresses, the frame-terminate address and the frame-continue address.

Table 2 Register Addressing

Register Selected	R/$\overline{W}$	RS_1	RS_0	Address Control Bit (C1b0)
Control Register 1	L	L	L	X
Control Register 2	L	L	H	L
Control Register 3	L	L	H	H
Transmitter Data FIFO Register (Frame Continue)	L	H	L	X
Transmitter Data FIFO Register (Frame Terminate)	L	H	H	L
Control Register 4	L	H	H	H
Status Register 1	H	L	L	X
Status Register 2	H	L	H	X
Receiver Data FIFO Register	H	H	X	X

H = HIGH Voltage Level
L = LOW Voltage Level
X = Don't Care

Receiver Data First-In First-Out Register (Rx FIFO)
The Rx FIFO consists of three 8-bit registers which are used for the buffer storage of received data. Data bytes are always transferred from a full register to an adjacent empty register; both phases of the E input clock are used for the data transfer. Each register has pointer bits which point the frame boundary. When these pointers appear at the last Rx FIFO location, they update the AP, FV or ERR status bits.

The RDA status bit indicates the state of the Rx FIFO. When RDA status bit is HIGH, the Rx FIFO is ready to be read. The RDA status is controlled by the 2/1-Byte control bit. When overrun occurs, the data in the first byte of the Rx FIFO are no longer valid.

Both the RxRS control and $\overline{RESET}$ input clear the Rx FIFO. Abort (in frame) and a HIGH level of $\overline{DCD}$ input also clear the Rx FIFO, but the last bytes of the previous frame, which are separated by the frame boundary pointer, are not disturbed.

Transmitter Data First-In First-Out Register (Tx FIFO)
The Tx FIFO consists of three 8-bit registers which are used for buffer storage of data to be transmitted. Data is always transferred from a full register to an empty adjacent register; the transfer occurs on both phases of the E input clock. The Tx FIFO can be addressed by two different register addresses, the frame-continue address and the frame-terminate address. Each register has pointer bits which point to the frame boundary. When a data byte is written at the frame-continue address, the pointer of the first FIFO register is set. When a data byte is written at the frame-terminate address, the pointer of the first FIFO register is reset. The RxRS or ABT control bit resets all pointers. The pointer will shift through the Tx FIFO. When a positive transition is detected at the third FIFO register, the Transmitter initiates a frame with an open flag. When the negative transition is detected at the third FIFO register, the Transmitter closes a frame, appending the FCS and closing flag to the last byte.

The Tx Last control bit can be used instead of using the frame-terminate address. When the Tx Last control bit is set with a HIGH, the logic searches the last byte location in the Tx FIFO and resets the pointer.

The status of Tx FIFO is indicated by the TDRA/FC status bit. When TDRA/FC is HIGH, the Tx FIFO is available for loading data. The TDRA/FC status is controlled by the 2/1-Byte control bit. The Tx FIFO is reset by both TxRS and the $\overline{RESET}$ input. During this reset condition or when $\overline{CTS}$ input is HIGH, the TDRA/FC status bit is suppressed and data loading is inhibited.

ADLC Internal Register Structure

		RS_1, RS_0 = LL	RS_1, RS_0 = LH	RS_1, RS_0 = HL	RS_1, RS_0 = HH
	Bit No.	Status Register 1	Status Register 2	Receiver Data FIFO Register	Unused
Read-Only Registers	0	Receive Data Available (RDA)	Address Present (AP)	Bit 0	Same as RS_1, RS_0 = HL
	1	Status Register 2 Read Request (S2RQ)	Frame Valid (FV)	Bit 1	
	2	Loop Status (LOOP)	Inactive Idle Received (RxIdle)	Bit 2	
	3	Flag Detected (FD)	Abort Received (Rx ABT)	Bit 3	
	4	Clear-to-Send (CTS)	Frame Check Sequence/Invalid Frame Error (ERR)	Bit 4	
	5	Transmitter Underrun (TxU)	Data Carrier Detect (DCD)	Bit 5	
	6	Transmitter Data Register Available/ Frame Complete (TDRA/FC)	Receiver Overrun (OVRN)	Bit 6	
	7	Interrupt Request (IRQ)	Received Data Available (RDA)	Bit 7	

					Transmitter Data FIFO Register		
	Bit No.	Control Register 1	Control Register 2 (C1b0 = L)	Control Register 3 (C1b0 = H)	Frame Continue	Frame Terminate (C1b0 = L)	Control Register 4 (C1b0 = H)
Write-Only Registers	0	Address Control (AC)	Prioritized Status Enable (PSE)	Logic Control Field Select (LCF)	Bit 0	Bit 0	Double Flag/Single Flag Interframe Control ("FF"/"F")
	1	Receiver Interrupt Enable (RIE)	2-Byte/1-Byte Transfer (2/1-Byte)	Extended Control Field Select (C_{EX})	Bit 1	Bit 1	Transmitter 1 Word Length Select ($TxWLS_1$)
	2	Transmitter Interrupt Enable (TIE)	Flag/Mark Idle Select (F/M Idle)	Auto/Address Extend Mode (A_{EX})	Bit 2	Bit 2	Transmitter 2 Word Length Select ($TxWLS_2$)
	3	Receiver Data Service Request Mode (RDSR Mode)	Frame Complete/ TDRA Select (FC/TDRA Select)	01/11 Idle (01/11 Idle)	Bit 3	Bit 3	Receiver 1 Word Length Select ($RxWLS_1$)
	4	Transmitter Data Service Request Mode (TDSR Mode)	Transmit Last Data (Tx Last)	Flag Detect Status Enable (FDSE)	Bit 4	Bit 4	Receiver 2 Word Length Select ($RxWLS_2$)
	5	Rx Frame Discontinue (DISCONTINUE)	Clear Receiver Status (CLR RxST)	Loop/Non-Loop Mode (LOOP)	Bit 5	Bit 5	Transmit Abort (ABT)
	6	Receiver Reset (RxRS)	Clear Transmitter Status (CLR TxST)	Go Active on Poll/Test (GAP/TST)	Bit 6	Bit 6	Abort Extend (ABT_{EX})
	7	Transmitter Reset (TxRS)	Request-to-Send Control (RTS)	Loop On-Line Control/DTR Control (LOC/DTR)	Bit 7	Bit 7	NRZI (Zero Complement)/NRZ Select (NRZI/NRZ)

Control Register 1 (CR1)

RS_1	RS_0	R/$\overline{W}$	AC	7	6	5	4	3	2	1	0
L	L	L	X	TxRS	RxRS	DISCONTINUE	TDSR Mode	RDSR Mode	TIE	RIE	AC

b0 Address Control (AC) — AC provides another register select signal internally. The AC bit is used in conjunction with the RS_0, RS_1 and R/$\overline{W}$ inputs to select particular registers, as shown in *Table 2.*

b1 Receiver Interrupt Enable (RIE) — RIE enables/disables the interrupt request caused by the receiver section (HIGH = enable, LOW = disable).

b2 Transmitter Interrupt Enable (TIE) — TIE enables/disables the interrupt request caused by the Transmitter (HIGH = enable, LOW = disable).

b3 Receiver Data Service Request Mode (RDSR Mode) RDSR Mode provides the capability of operation with a bus system in the DMA mode when used in conjunction with the prioritized status mode. When RDSR Mode is set, an interrupt request caused by the RDA status bit is inhibited, and the ADLC does not request data transfer via the $\overline{IRQ}$ output.

b4 Transmitter Data Service Request Mode (TDSR Mode) — TDSR Mode provides the capability of operation with a bus system in the DMA mode when used in conjunction with the prioritized status mode. When TDSR Mode is set, an interrupt request caused by the TDRA/FC status bit is inhibited, and the ADLC does not request a data transfer via the $\overline{IRQ}$ output.

b5 Rx Frame Discontinue (DISCONTINUE) — When DISCONTINUE is set, the currently received frame is ignored and the ADLC discards the data of the current frame. DISCONTINUE is automatically reset when the last byte of the frame is discarded or when the ignored frame is aborted by receiving an abort or $\overline{DCD}$ failure.

b6 Receiver Reset (RxRS) — When RxRS is HIGH, the receiver section stays in the reset condition. All receiver sections, including the Rx FIFO and the receiver status bits in both status registers, are reset. (During reset, the stored DCD status is reset but the DCD status bit follows the $\overline{DCD}$ input.) RxRS is set by forcing a LOW level on the $\overline{RESET}$ input or by writing a HIGH into RxRS from the data bus. RxRS will be reset by writing a LOW from the data bus after $\overline{RESET}$ has gone HIGH.

b7 Transmitter Reset (TxRS) — When TxRS is HIGH, the transmitter section stays in the reset condition and transmits marks (1s). All transmitter sections, including the Tx FIFO and the transmitter status bits, are reset (Tx FIFO cannot be loaded). During reset, the stored $\overline{CTS}$ status is reset but the CTS status bit follows the $\overline{CTS}$ input. TxRS is set by forcing a LOW level on the $\overline{RESET}$ input or by writing a HIGH from the data bus. It will be reset by writing a LOW after $\overline{RESET}$ has gone HIGH.

5

Control Register 2 (CR2)

RS_1	RS_0	R/$\overline{W}$	AC	7	6	5	4	3	2	1	0
L	H (C1b0 = L)	L	L	RTS	CLR TxST	CLR RxST	Tx Last	FC/TDRA Select	F/M Idle	2/1 Byte	PSE

b0 Prioritized Status Enable (PSE) — When PSE is set, the status bits in both status registers are prioritized as defined in the Status Register section. When PSE is LOW, the status bits indicate current status without bit suppression by other status bits. The exception to this rule is the CTS status bit which always suppresses the TDRA/FC status bit.

b1 2-Byte/1-Byte Transfer (2/1-Byte) — When 2/1-Byte is reset, the TDRA/FC and RDA status bits then will indicate the availability of their respective data FIFO registers for a single-byte data transfer. Similarly, if 2/1-Byte is set, the TDRA/FC and RDA status bits indicate when two bytes of data can be moved without a second status read.

b2 Flag/Mark Idle Select (F/M Idle) — F/M Idle Select selects flag characters or bit-by-bit mark idle for the time fill or the idle state of the Transmitter. When mark idle is selected, go-ahead code can be generated for loop operation in conjunction with the 01/11 Idle control bit (HIGH = flag time fill, LOW = mark idle).

b3 Frame Complete/TDRA Select (FC/TDRA Select) — FC/TDRA Select selects TDRA status or FC status for the TDRA/FC status bit indication (HIGH = FC status, LOW = TDRA status).

b4 Transmit Last Data (Tx Last) — Tx Last provides another method to terminate a frame. When the Tx Last is set just after loading a data byte, the ADLC assumes the byte is the last byte and terminates the frame by appending Cyclic Redundancy Check Character (CRCC) and a closing flag. This control bit is useful for DMA operation. Tx Last automatically returns to the LOW state.

b5 Clear Receiver Status (CLR RxST) — When a HIGH is written into CLR RxST, a reset signal is generated for the receiver status bits in Status Registers 1 and 2 (except AP and RDA bits). The reset signal is enabled only for the bits which have been present during the last read status operation. CLR RxST automatically returns to the LOW state.

b6 Clear Transmitter Status (CLR TxST) — When a HIGH is written into CLR TxST, a reset signal is generated for the transmitter status bits in Status Register 1 (except TDRA/FC). The reset signal is enabled for the bits which have been present during the last read status operation. CLR TxST automatically returns to the LOW state.

b7 Request-to-Send Control (RTS) — RTS, when HIGH, causes the $\overline{RTS}$ output to be LOW (the active state). When the RTS bit returns LOW and data is being transmitted, the $\overline{RTS}$ output remains LOW until the last character of the frame (the closing flag or abort) has been completed and the Tx FIFO is empty. If the Transmitter is idling when the RTS bit returns LOW, the $\overline{RTS}$ ouput will go HIGH (the inactive state) within two bit-times.

Control Register 3 (CR3)

RS_1 L RS_0 H $R/\overline{W}$ L AC H (C1b0 = H)

7	6	5	4	3	2	1	0
LOC/DTR	GAP/TST	LOOP	FDSE	01/11 Idle	A_{EX}	C_{EX}	LCF

b0 Logic Control Field Select (LCF) — LCF causes the first byte(s) of data belonging to the information field to remain 8-bit characters until the logic control field is complete. The logic control field (when selected) is an automatically extendable field which is extended when bit 7 of a logic control character is HIGH. When LCF is reset, the ADLC assumes no logic control field is present for either the transmitted or received data channels. When the logic control field is terminated, the word length of the information data is then defined by Rx or Tx WLS_1 and WLS_2.

b1 Extended Control Field Select (C_{EX}) — When the C_{EX} bit is HIGH, the control field is extended and assumed to be 16 bits. When C_{EX} is LOW, the control field is assumed to be eight bits.

b2 Auto/address Extend Mode (A_{EX}) — A_{EX}, when LOW, allows a full eight bits of the address octet to be utilized for addressing, because address extension is inhibited. When A_{EX} is HIGH, bit 0 of address octet equal to 0 causes the address field to be extended by one octet. The exception to this automatic address field extension is when the first address octet is all 0s (the null address).

b3 01/11 Idle (01/11 Idle) — The 01/11 Idle control bit determines whether the inactive (mark) idle condition begins with a 0 or not. If 01/11 Idle is set, the closing flag (or abort) will be followed by a 011111... pattern. This is required of the controller for the go-ahead character in the loop mode. When 01/11 Idle is reset, the idling condition will be all 1s.

b4 Flag Detect Status Enable (FDSE) — FDSE enables the Flag Detected (FD) status bit in Status Register 1 to indicate the occurrence of a received flag character. The status indication will be accompanied by an interrupt if the RIE control bit is set. Flag detection will cause the $\overline{FLAG}$ $\overline{DET}$ output to go LOW for one bit-time regardless of the state of FDSE.

b5 Loop/Non-Loop Mode (LOOP) — When LOOP is set, loop mode operation is selected and the GAP/TST control bit, LOC/DTR control bit and $\overline{LOC}/\overline{DTR}$ output are selected to perform the loop control functions. When LOOP is reset, the ADLC operates in the point-to-point data communications mode.

b6 Go Active on Poll/Test (GAP/TST) — In the loop mode GAP/TST is used to respond to the poll sequence and to begin transmission. When GAP/TST is set, the Receiver searches for the go-ahead (or end-of-poll, EOP). The Receiver go-ahead is converted to an opening flag and the ADLC starts its own transmission. When GAP/TST is reset during the transmission, the end of the frame (the completion of flag or abort) causes the termination of the go-active-on-poll operation and the RxData to TxData link is reestablished. The ADLC then returns to the loop-on-line state.

In the non-loop mode GAP/TST is used for self-test purposes. If GAP/TST is set, the TxData output is connected to the RxData input internally, and provides a loop-back feature. For normal operation, the GAP/TST bit should be reset.

b7 Loop On-line Control/DTR Control (LOC/DTR) — In the loop mode LOC/DTR is used to go on-line or to go off-line. When LOC/DTR is set, the ADLC goes to the on-line state after seven consecutive 1s occur at the RxData input. When LOC/DTR is reset, the ADLC goes to the off-line state after eight consecutive 1s occur at the RxData input.

In the non-loop mode the LOC/DTR bit directly controls the $\overline{LOC}/\overline{DTR}$ output state (HIGH = $\overline{DTR}$ output goes to LOW level, LOW = $\overline{DTR}$ output goes to HIGH level).

5

Control Register 4 (CR4)

RS_1	RS_0	$R/\overline{W}$	AC	7	6	5	4	3	2	1	0
H	H (C1b0 = H)	L	H	NRZI/NRZ	ABT_{EX}	ABT	$RxWLS_2$	$RxWLS_1$	$TxWLS_2$	$TxWLS_1$	"FF"/"F"

b0 Double Flag/Single Flag Interframe Control ("FF"/"F") — "FF"/"F" determines whether the Transmitter will transmit separate closing and opening flags when frames are transmitted successively. When the "FF"/"F" control bit is LOW, the closing flag of the first frame will serve as the opening flag of the second frame; when HIGH, independent opening and closing flags will be transmitted.

b1, b2 Transmitter Word Length Select ($TxWLS_1$, $TxWLS_2$)- $TxWLS_1$ and $TxWLS_2$ are used to select the word length of the Transmitter information field. The encoding format is shown in *Table 3.*

b3, b4 Receiver Word Length Select ($RxWLS_1$, $RxWSL_2$) — $RxWLS_1$ and $RxWLS_2$ are used to select the word length of the Receiver information field. The encoding format is shown in *Table 3.*

b5 Transmit Abort (ABT) — ABT causes an abort (at least eight bits of 1 in succession) to be transmitted. The abort is initiated and the Tx FIFO is cleared when ABT goes HIGH. Once abort begins, the ABT bit assumes the LOW state.

b6 Abort Extended (ABT_{EX}) — If ABT_{EX} is set, the abort code initiated by ABT is extended at least 16 bits of consecutive 1s, the mark idle state.

b7 NRZI (Zero Complement)/NRZ Select (NRZI/NRZ) NRZI/NRZ selects the transmit/receive data format to be NRZI or NRZ in both loop mode or non-loop mode operation. When the NRZI mode is selected, a 1-bit delay is added to the transmitted data (TxData) to allow for NRZI encoding (HIGH = NRZI, LOW = NRZ).

Note
NRZI coding — The serial data remains in the same state to send a binary 1 and switches to the opposite state to send a binary 0.

Table 3 I-Field Character Length Select

WLS_1	WLS_2	I-Field Character Length
L	L	5 bits
H	L	6 bits
L	H	7 bits
H	H	8 bits

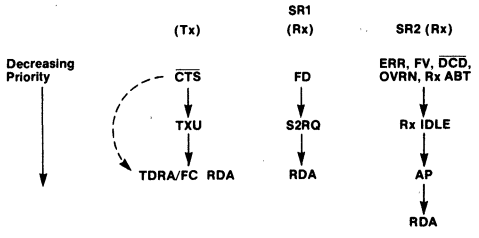
Status Registers
Status Register 1 is the main status register. The $\overline{IRQ}$ bit indicates whether the ADLC requests service or not. The S2RQ bit indicates whether any bits in Status Register 2 request service. TDRA/FC and RDA, because they are most often used, are located in bit positions that are more convenient to test. RDA reflects the state of the RDA bit in Status Register 2.

Status Register 2 provides the detailed status information contained in the S2RQ bit, and these bits reflect receiver status.

The prioritized status mode provides maximum efficiency in searching the status bits and indicates only the most important action required to service the ADLC. The priority trees of both status registers are provided in *Figure 3.*

Reading the status register is a non-destructive process. The method of clearing status depends upon the bit function and is discussed for each bit in the register.

Fig. 3 Status Register Priority Tree (PSE = 1)



* Prioritized even when PSE = 0

Note
Status bit above will inhibit one below it.

Status Register 1 (SR1)

RS_1	RS_0	R/$\overline{W}$	AC	7	6	5	4	3	2	1	0
L	L	H	X	IRQ	TDRA/FC	TXU	CTS	FD	LOOP	S2RQ	RDA

b0 Receiver Data Available (RDA) — The RDA status bit reflects the state of the RDA status bit in status Register 2. It provides the means of achieving data transfers of received data in the full-duplex mode without having to read both status registers.

b1 Status Register 2 Read Request (S2RQ) — All the status bits (stored conditions) of Status Register 2 (except RDA) are logically ORed and indicated by the S2RQ status bit. Therefore S2RQ indicates when Status Register 2 needs to be read. When S2RQ is LOW, it is not necessary to read Status Register 2. The bit is cleared when the appropriate bits in Status Register 2 are cleared or when RxRS is used.

b2 Loop Status (LOOP) — The LOOP status bit is used to monitor the loop operation of the ADLC. This bit does not cause an IRQ. When non-loop mode is selected, LOOP stays LOW; when loop mode is selected, the LOOP goes to HIGH during on-loop condition. When ADLC is in an off-loop condition or go-active-on-poll condition, the LOOP status bit is LOW.

b3 Flag Detected (FD) — The FD status bit indicates that a flag has been received if the Flag Detect Status Enable control bit has been set. FD goes HIGH at the last bit of the flag character received (when the $\overline{FLAG}$ $\overline{DET}$ output goes LOW) and is stored until cleared by clear RxST or RxRs.

b4 Clear-To-Send (CTS) — The $\overline{CTS}$ input positive transition is stored in Status Register 1 and causes an IRQ (if enabled). The stored CTS condition and its IRQ are cleared by CLR TxST or TxRS control bit. After the stored status is reset, the CTS status bit reflects the state of the $\overline{CTS}$ input.

b5 Transmitter Underrun (TxU) — When the transmitter runs out of data during a frame transmission, an underrun occurs and the frame is automatically terminated by transmitting an abort. The underrun condition is indicated by the TxU status bit. TxU can be cleared by means of the CLK TxST control bit or by TxRS.

b6 Transmitter Data Register Available/Frame Complete (TDRA/FC) — The TDRA/FC status bit serves two purposes, depending upon the state of the FC/TDRA Select control bit. When TDRA/FC serves as a TDRA status bit, it indicates that data (to be transmitted) can be loaded into the Tx FIFO. The first register (Register 1) of the Tx FIFO being empty (TDRA = HIGH) will be indicated by the TDRA/FC status bit in the 1-byte transfer mode. The first two registers (Registers 1 and 2) must be empty for TDRA to be HIGH when in the 2-byte transfer mode. TDRA/FC is inhibited by TxRS or $\overline{CTS}$ being HIGH.

When the frame-complete mode of operation is selected, the TDRA/FC status bit goes HIGH when an abort is transmitted or when a flag is transmitted with no data in the Tx FIFO. The bit remains HIGH until cleared by resetting the FC/TDRA Select or setting the TxRS control bit.

b7 Interrupt Request (IRQ) — The Interrupt Request status bit indicates when the $\overline{IRQ}$ output is in the active state ($\overline{IRQ}$ output = LOW). The IRQ status bit is subject to the same interrupt enables (RIE, TIE) as the $\overline{IRQ}$ output. The IRQ status bit simplifies status inquiries for polling systems by providing single-bit indication of service requests.

5

Status Register 2 (SR2)

RS_1	RS_0	R/$\overline{W}$	AC
L	H	L	X

7	6	5	4	3	2	1	0
RDA	OVRN	DCD	ERR	RxABT	Rx Idle	FV	AP

b0 Address Present (AP) — The AP status bit provides the frame boundary and indicates an address octet is available in the Rx FIFO. In the extended addressing mode, the AP bit continues to indicate addresses until the address field is complete. The AP status bit is cleared by reading data or by RxRS.

b1 Frame Valid (FV) — The FV status bit provides the frame boundary indication to the MPU and also indicates that a frame is complete with no error. The FV status bit is set when the last data byte of a frame is transferred into the last location of the Rx FIFO (available to be read by MPU). Once FV status is set, the ADLC stops further data transfer into the last location of the Rx FIFO (in order to prevent the mixing of two frames) until the status bit is cleared by the CLR RxST or RxRS control bit.

b2 Inactive Idle Received (Rx Idle) — The Rx Idle status bit indicates that a minimum of 15 consecutive 1s have been received. The event is stored within the status register and can cause an interrupt. The interrupt and stored condition are cleared by the CLR RxST control bit. Rx Idle is the logical OR of the receiver idling detector (which continues to reflect idling until a LOW is received) and the stored inactive idle condition.

b3 Abort Received (RxABT) — The RxABT status bit indicates that seven or more consecutive 1s have been received. Abort has no meaning under out-of-frame conditions; therefore, no interrupt or storing of the status will occur unless a flag has been detected prior to the abort. An abort received when in-frame is stored in the status register and causes an $\overline{IRQ}$. The RxABT is the logical OR of the stored conditions and the receiver abort detect logic, which is cleared after 15 consecutive 1s have occurred. The stored abort condition is cleared by the CLR RxST or RxRS control bit.

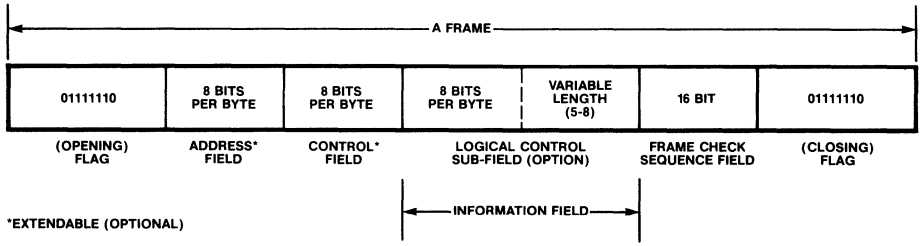
b4 Frame Check Sequence/Invalid Frame Error (ERR) When a frame is complete with a cyclic redundancy check (CRC) error or a short frame error (the frame does not have complete address and control fields), the ERR status bit is set instead of the Frame Valid status bit. Other functions, frame boundary indication and control function, are exactly the same as for the Frame Valid status bit. Refer to the FV status bit.

b5 Data Carrier Detect (DCD) — A positive transition on the $\overline{DCD}$ input is stored in the status register and causes an IRQ (if enabled). The stored DCD condition and its IRQ are cleared by the CLR RxST control bit or RxRS. After stored status is reset, the DCD status bit follows the state of the $\overline{DCD}$ input. Both the stored DCD condition and the $\overline{DCD}$ input cause the reset of the receiver section when they are HIGH.

b6 Receiver Overrun (OVRN) — The OVRN status bit indicates that receiver data has been transferred into the Rx FIFO when it is full, resulting in data loss. The OVRN status bit is cleared by the CLR RxST or RxRS control bit. Continued overrunning only destroys data in the first FIFO register.

b7 Receiver Data Available (RDA) — The Receiver Data Available status bit indicates when receiver data can be read from the Rx FIFO. When the prioritized status mode is used, the RDA bit indicates that non-address and non-last data are available in the Rx FIFO. The receiver data being present in the last register of the FIFO causes RDA to be HIGH for the 1-byte transfer mode. The RDA bit being HIGH indicates that the last two registers are full when in the 2-byte transfer mode. The RDA status bit is reset automatically when data is not available.

Frame Format



The ADLC transmits and receives data (information or control) in a format called a frame. All frames start with an opening flag (F) and end with a closing flag (F). Between the opening flag and closing flag, a frame contains an address field, control field, information field, and frame check sequence field.

Flag (F)
The flag is the unique binary pattern 01111110. It provides the frame boundary and a reference for the position of each field of the frame.

The ADLC Transmitter generates a flag pattern internally and the opening flag and closing flags are appended to a frame automatically. Two successive frames can share one flag for a closing flag of the first frame and for the opening flag of the next frame, if the "FF/F" control bit in Control Register 4 is reset.

The Receiver searches for a flag on a bit-by-bit basis and recognizes a flag at any time. The Receiver establishes the frame synchronization with every flag. The flags mark the frame boundary and reference for each field but they are not transferred to the Rx FIFO. The detection of a flag is indicated by the $\overline{\text{FLAG}}$ $\overline{\text{DET}}$ output and by the FD Status bit.

Order of Bit Transmission
Address, control and information field bytes are transferred between the MPU and the ADLC in parallel by means of the data bus. The bit on D_0 (data bus bit 0, pin 22) is serially transmitted first, and the first serially received bit is transferred to the MPU on D_0. The FCS field is transmitted and received MSB first.

Address (A) Field
The eight bits following the opening flag are the address (A) field. The A-field can be extendable if the auto-address extend mode is selected in Control Register 3. In the address extend mode, the first bit (bit 0) in every address octet becomes the C_{EX} control bit. When the bit is LOW, the ADLC assumes another address octet will follow; when the bit is HIGH, the address extension is terminated. A "null" address (all LOW) does not extend. In the Receiver, the AP status bit distinguishes the address field from other fields. When an address byte is available to be read in the Rx FIFO, the AP status bit is set and causes an interrupt (if enabled). The AP status bit is set for every addresss octet when the address extend mode is used.

Control (C) Field
The eight bits following the address field are the control (link control) field. When the C_{EX} control bit in Control Register 3 is selected, the C-field is extended to 16 bits.

Information (I) Field
The I-field follows the C-field and precedes the FCS field. The I-field contains data to be transferred but is not always necessarily contained in every frame. The word length of the I-field can be selected from five to eight bits per byte by control bits in Control Register 4. The I-field will continue until it is terminated by the FCS and closing flag. The Receiver has the capability to handle a partial last byte. The last information byte can be any word length between one and eight bits. If the last byte in the I-field is less than the selected word length, the Receiver will right justify the received bits, fill the remaining bits of the receiver shift register with zeros, and transfer a full byte to the Rx FIFO. Regardless of selected byte length, the ADLC will transfer eight bits of data to the data bus. Unused bits for word lengths of five, six, and seven will be zeroed.

Logic Control (LC) Field
When the LCF control bit in Control Register 3 is selected, the ADLC separates the I-field into two sub-fields. The first sub-field is the logic control field and the following sub-field is the data portion of the I-field. The logic control field is eight bits and follows the C-field, which is extendable by

octets, if it is selected. The last bit (bit 7) is the C_{EX} bit, and if it is HIGH, the LC-field is extended one octet.

Note

Hereafter, the term information field, or I-field, is used as the data portion of the information field and excludes the logic control field. This is done in order to keep the consistency of the meaning of information field as specified in SDLC, HDLC, and ADCCP standards.

Frame Check Sequence (FCS) Field

The 16 bits preceding the closing flag are the FCS field. The FCS is the cyclic redundancy check character (CRCC). The polynomial $x^{16} + x^{12} + x^5 + 1$ is used both for the Transmitter and Receiver. Both the transmitter and receiver polynomial registers are initialized to all 1s prior to calculation of the FCS. The Transmitter calculates the FCS on all bits of the address, control, logic control (if selected), and information fields, and transmits the complement of the resulting remainder as FCS. The Receiver performs a similar computation on all bits of the address, control, logic control (if selected), information, and received FCS fields and compares the result to FOB8 (hexadecimal). When the result matches FOB8, the FV status bit is set in Status Register 2. If the result does not match, the ERR status bit is set. The FCS generation, transmission, and checking are performed automatically by the ADLC Transmitter and Receiver. The FCS field is not transferred to the Rx FIFO.

Invalid Frame

Any valid frames should have at least the A-field, C-field and FCS field between the opening flag and the closing flag. When invalid frames are received, the ADLC handles them as follows:

1. A short frame which has less than 25 between flags—The ADLC ignores the short frame and its reception is not reported to the MPU.
2. A frame less than 32 bits between the flags, or a frame 32 bits or more with an extended A-field or C-field that is not completed is transferred into the Rx FIFO. The ERR status bit indicates the reception of the invalid frame at the end of the frame.
3. Aborted frame—The frame which is aborted by receiving an abort or $\overline{DCD}$ failure is also an invalid frame. Refer to ABT and DCD status bits.

Zero Insertion and Zero Deletion

The zero insertion and deletion that allows the content of the frame to be transparent is performed by the ADLC automatically. A binary 0 is inserted by the Transmitter after any succession of five 1s within a frame (A, C, LC, I, and FCS field). The Receiver deletes a binary 0 that follows five successive 1s within a frame.

Abort

The function of prematurely terminating a data link is called abort. The Transmitter aborts a frame by sending at least eight consecutive 1s immediately after the ABT control bit in Control Register 4 is set to HIGH. (Tx FIFO is also cleared by the ABT control bit at the same time.) The abort can be extended to at least 16 consecutive 1s, if the ABT_{EX} control bit in the Control Register 4 is set when an abort is sent. This feature is useful to force mark idle transmission. Reception of seven or more consecutive 1s is interpreted as an abort by the Receiver. The Receiver responds to a received abort as follows:

1. An abort in an out-of-frame condition—An abort during the idle or time fill has no meaning. The abort reception is indicated in the Status Register as long as the abort condition continues, but neither an interrupt nor a stored condition occurs. The abort indication disappears after 15 or more consecutive 1s are received (Rx Idle status bit is set).
2. An abort in frame, when less than 26 bits have been received after an opening flag, has not transferred any field to the MPU. The ADLC clears the aborted frame data in the Rx FIFO and clears flag synchronization. Neither an interrupt nor a stored status occurs. The status indication is the same as (1) above.
3. An abort in-frame, when 26 bits or more have been received after an opening flag, might have transferred some fields of the aborted frame onto the data bus. The abort status is stored in Status Register 2 and the data of the aborted frame in the ADLC is cleared. The synchronization is also cleared.

Idle and Time Fill

When the Transmitter is in an out-of-frame condition (the Transmitter is not transmitting a frame), it is in an idle state. Either a series of contiguous flags (time fill) or a mark idle (consecutive 1s on a bit-by-bit basis) is selected for the transmission in an idle state by the F/M Idle control bit. When the Receiver receives 15 or more consecutive 1s, the Rx Idle status bit is set and causes an interrupt. The flags and mark idle are not transferred to the Rx FIFO.

Programming Considerations

1. **Status Priority** — When the prioritized status mode is used, it is best to test for the lowest priority conditions first. The lowest priority conditions typically occur more frequently and are the most likely conditions to exist when the processor is interrupted.

2. **Stored vs. Present Status** — Certain status bits (DCD, CTS, RxABT, and Rx Idle) indicate a status which is the logic OR of a stored and a present condition. It is the stored status that causes an interrupt and which is

cleared by the CLR RxST or CLR TxST control bit. After being cleared, the status register will reflect the present condition of an input or a receiver input sequence.

3. **Clearing Status Registers** — In order to clear an interrupt with the two status clear control bits, a particular status condition must be read before it can be cleared. In the prioritized mode, clearing a higher priority condition might result in another $\overline{IRQ}$ caused by a lower priority condition whose status was suppressed when a status register was first read. This guarantees that a status condition is never inadvertently cleared.

4. **Clearing the Rx FIFO** — An RxRS will effectively clear the contents of all three Rx FIFO bytes. However, the Rx FIFO may contain data from two different frames when abort or DCD failure occurs. When this happens, the data from a previously closed frame (a frame whose closing flag has been received) will not be destroyed.

5. **Servicing the Rx FIFO in a 2-Byte Mode** — The procedure for reading the last bytes of data is the same, regardless of whether the frame contains an even or an odd number of bytes. Continue to read two bytes until an interrupt occurs that is caused by an end-of-frame status (FV or ERR). When this occurs, indicating that the last byte either has been read or is ready to be read, switch temporarily to the 1-byte mode with no prioritized status (Control Register 2). Test RDA to indicate whether a 1-byte read should be performed. Then clear the frame end status.

6. **Frame Complete Status and RTS Release** — In many cases, a modem will require a delay for releasing $\overline{RTS}$. An 8-bit or 16-bit delay can be added to the ADLC $\overline{RTS}$ output by using an abort. At the end of a transmission, frame-complete status will indicate the frame completion. After the TDRA/FC status bit goes HIGH, write 1 into the ABT control bit (and ABT_{EX} bit if a 16-bit delay is required). After the ABT control bit is set, write 0 into the RTS control bit. The Transmitter will transmit eight or 16 1s and the $\overline{RTS}$ output will then go HIGH (inactive).

7. **Note to users not using the F6800** — (a) Care should be taken when performing a write followed by a read on successive E pulses at a high frequency rate. Time must be allowed for status changes to occur. If this is done, the time that E is LOW between successive write/read E pulses should be at least 500 ns. (b) The ADLC is a completely static part. However, the E frequency should be high enough to move data through the FIFO registers and to service the peripheral requirements. Also, the period between successive E pulses should be less than the period of RxC or TxC in order to maintain synchronization between the data bus and the peripherals.

Absolute Maximum Ratings

Supply Voltage, V_{CC}	-0.3V, +7.0V
Input Voltage, V_{IN}	-0.3V, +7.0V
Operating Temperature, T_A	
F6854P/S, F68A54P/S, F68B54P/S	0° C, +70° C
F6854CP/CS, F68A54CP/CS	-40° C, +85° C
F6854DL	-55° C, +85° C
F6854DM	-55° C, +125° C
Storage Temperature, T_{STG}	-65° C, +150° C

Thermal Resistance, θ_{JA}	
Plastic	115° C/W
Ceramic	60° C/W

These devices contain circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit.

DC Characteristics V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = over operating temperature range, unless otherwise noted

Symbol	Characteristic	Min	Typ	Max	Unit	Test Condition
V_{IH}	Input HIGH Voltage	V_{SS} + 2.0			V	
V_{IL}	Input LOW Voltage			V_{SS} + 0.8	V	
V_{OH}	Output HIGH Voltage D_0-D_7 All Others	 V_{SS} + 2.4 V_{SS} + 2.4			V	 I_{Load} = -205 μA I_{Load} = -100 μA
V_{OL}	Output LOW Voltage			V_{SS} + 0.4	V	I_{Load} = 1.6 mA
I_{IN}	Input Leakage Current All Inputs Except D_0-D_7		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	Three-State (Off State) Input Current D_0-D_7		2.0	10	μA	V_{IN} = 0.4 to 2.4 V V_{CC} = 5.25 V
I_{LOH}	Output Leakage Current (Off State) $\overline{IRQ}$		1.0	10	μA	V_{OH} = 2.4 V
P_D	Power Dissipation			850	mW	V_{CC} = 5.25 V
C_{IN}	Input Capacitance D_0-D_7 All Other Inputs			 12.5 7.5	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance $\overline{IRQ}$ All Others			 5.0 10	pF	

AC Characteristics

Symbol	Characteristic	F6854 Min	F6854 Max	F68A54 Min	F68A54 Max	F68B54 Min	F68B54 Max	Unit
PW_{CL}	Minimum Clock Pulse Width, LOW	700		450		280		ns
PW_{CH}	Minimum Clock Pulse Width, HIGH	700		450		280		ns
f_{max}	Clock Frequency		0.66		1.0		1.5	MHz
t_{RDSU}	Receive Data Set-up Time	250		200		120		ns
t_{RDH}	Receive Data Hold Time	120		100		60		ns
t_{RTS}	Request-to-Send Delay Time		680		460		340	ns
t_{TDD}	Clock-to-Data Delay for Transmitter		460		320		250	ns
t_{FD}	Flag Detect Delay Time		680		460		340	ns
t_{DTR}	DTR Delay Time		680		460		340	ns
t_{LOC}	Loop On-line Control Delay Time		680		460		340	ns
t_{RDSR}	RDSR Delay Time		540		400		340	ns
t_{TDSR}	TDSR Delay Time		540		400		340	ns
t_{IR}	Interrupt Request Release Time		1.2		0.9		0.7	μs
t_{RES}	RESET Minimum Pulse Width	1.0		0.65		0.40		μs
t_r, t_f	Input Rise and Fall Times (Except Enable) 0.8 V to 2.0 V		1.0*		1.0*		1.0*	μs

* 1.0μs or 10% of the pulse width, whichever is smaller.

Bus Timing Characteristics V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = over operating temperature range, unless otherwise noted

Symbol	Characteristic	F6854		F68A54		F68B54		Unit
		Min	Max	Min	Max	Min	Max	
Read								
PW_{EH}	Enable Pulse Width, HIGH	0.45		0.28		0.22		μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.28		0.21		μs
t_{cycE}	Enable Cycle Time	1.0		0.666		0.50		μs
t_{AS}	Set-up Time, Address and R/W Valid to Enable Positive Transition	160		140		70		ns
t_{DDR}	Data Delay Time		320		220		180	ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input		25		25		25	ns
Write								
PW_{EH}	Enable Pulse Width, HIGH	0.45		0.28		0.22		μs
PW_{EL}	Enable Pulse Width, LOW	0.43		0.28		0.21		μs
t_{cycE}	Enable Cycle Time	1.0		0.666		0.50		μs
t_{AS}	Set Time, Address and R/W Valid to Enable Positive Transition	160		140		70		ns
t_{DSW}	Data Set-up Time	195		80		60		ns
t_H	Data Hold Time	10		10		10		ns
t_{AH}	Address Hold Time	10		10		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input		25		25		25	ns

Fig. 4 Bus Timing Test Loads

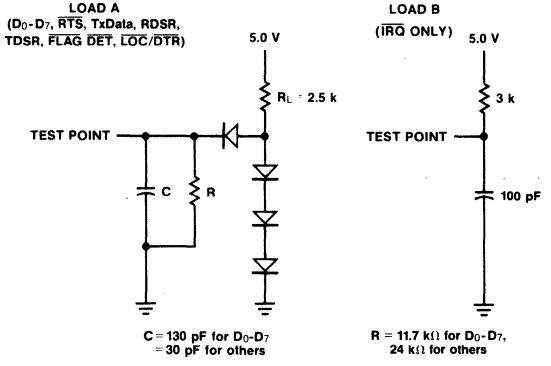


Fig. 5 Receiver Data Set-up/Hold, Flag Detect and Loop On-line Control Delay Timing

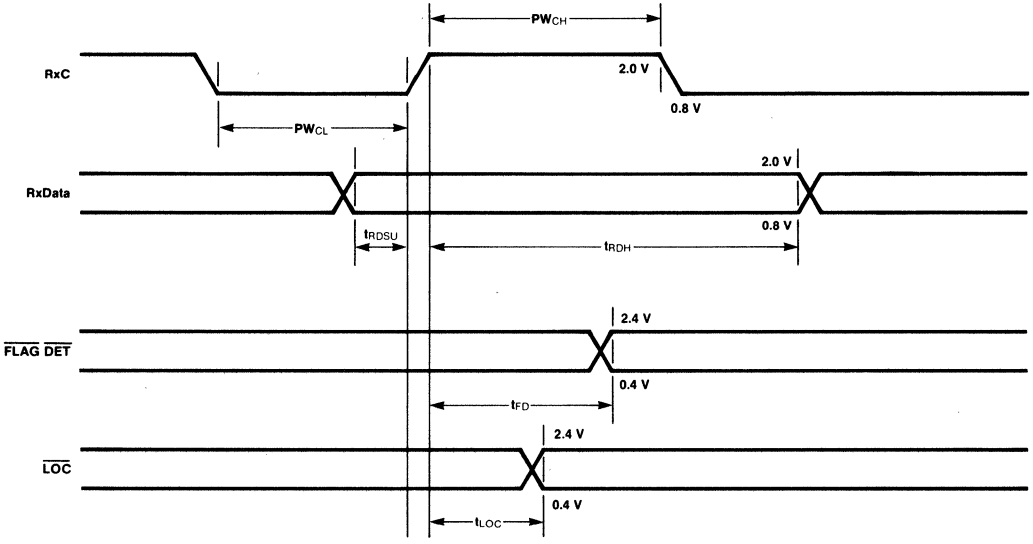


Fig. 6 Transmit Data Output Delay and Request-to-Send Delay Timing

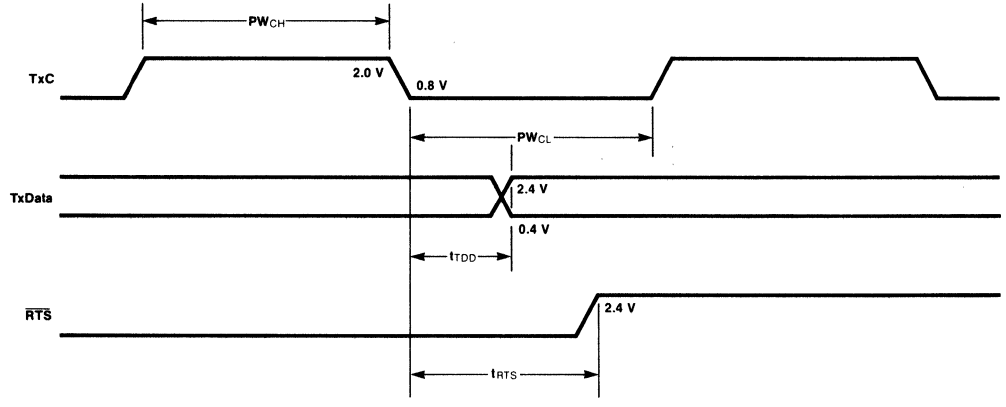


Fig. 7 TDSR/RDSR Delays, IRQ Release Delay, $\overline{RTS}$ and $\overline{DTR}$ Delay Timing

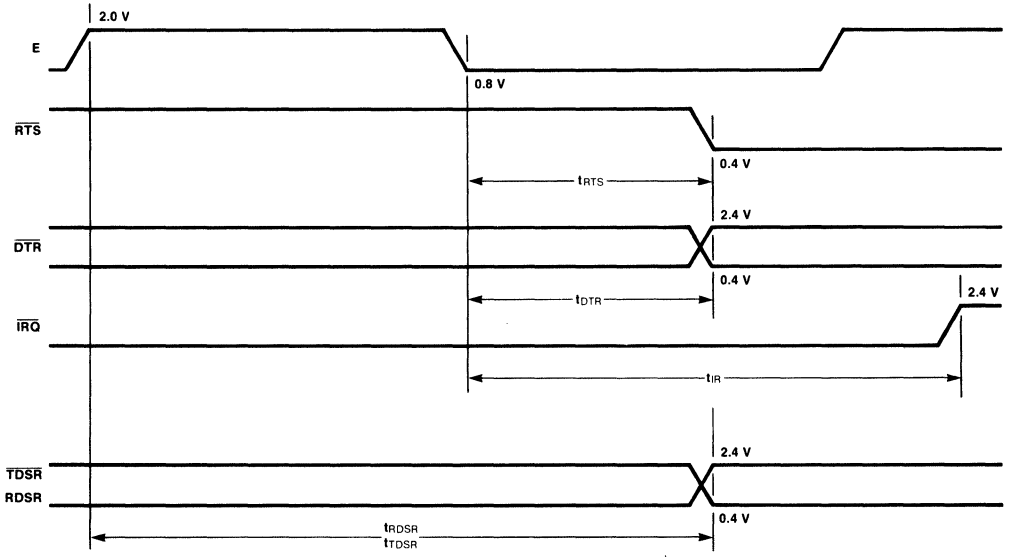
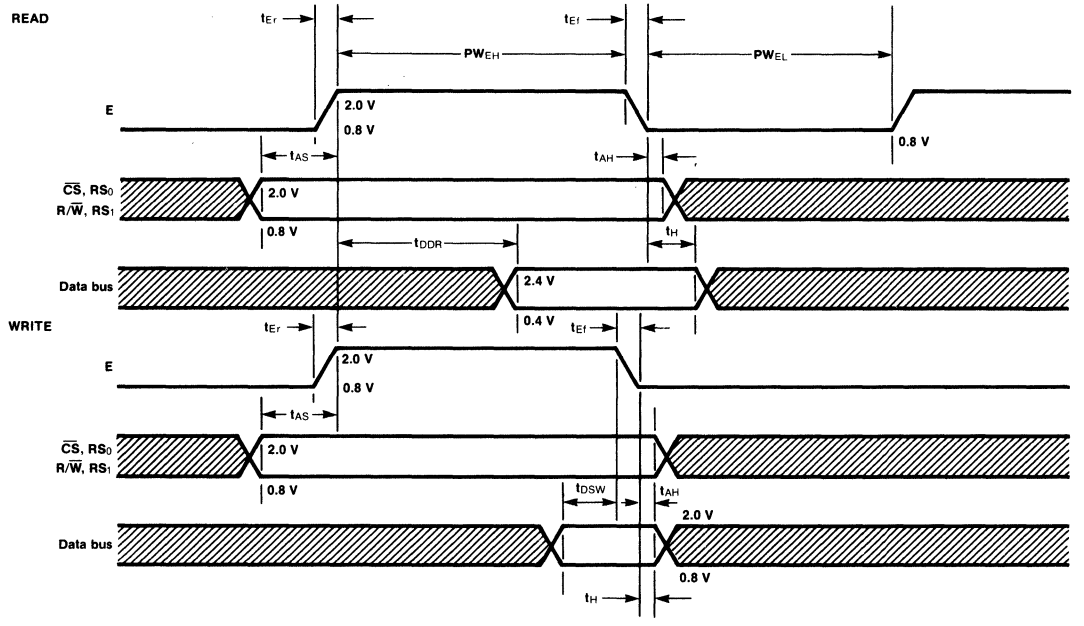


Fig. 8 Bus Read/Write Timing Characteristics



Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6854P,S	0 to +70°C
	F6854CP,CS	-40 to +80°C
	F6854DL	-55 to +80°C
	F6854DM	-55 to +80°C
1.5 MHz	F68A54P,S	0 to +70°C
	F68A54CP,CS	-40 to +80°C
2.0 MHz	F68B54P,S	0 to +70°C

P= Plastic package. S = Ceramic package

FAIRCHILD
A Schlumberger Company

F6856
Synchronous Protocol Communications Controller

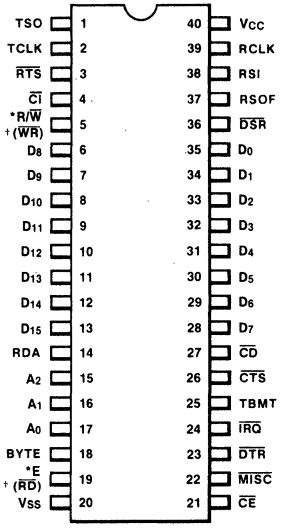
Microprocessor Product

Description

The F3846/F6856 Synchronous Protocol Communications Controller (SPCC) is a monolithic n-channel MOS LSI circuit designed to satisfy the major interface requirements of the bit-oriented protocol (BOP) and byte-control protocol (BCP). The SPCC converts parallel data from the CPU into a continuous serial data stream for transmission. Simultaneously, it converts received serial data to parallel data for the CPU. The SPCC is organized to interface with either an 8- or 16-bit bidirectional data bus, is fully TTL-compatible, and operates from a single +5 V supply.

- **F6800 and 8080 Bus Compatible**
- **Data Rate From DC to 1M BPS**
- **Bit-Oriented Line Control Protocols: SDLC, ADCCP, HDLC**
  - **Automatic Detection and Generation of Special Control Sequences (e.g., Flag, Abort, Go-Ahead)**
  - **Zero Insertion and Deletion**
  - **Primary or Secondary Station Select**
  - **Global Address**
  - **Automatic Extended Address**
  - **One or Two Control Bytes**
  - **Data Character Length From Five to Eight Bits with 1- to 8-Bit Residual Last Character**
  - **CCITT-CRC Error Detection**
  - **Interrupt on End of Message**
  - **IBM Retail Store Loop Mode**
- **Byte Control Protocol: IBM BISYNC**
  - **Special Character Generation: DLE, SYNC**
  - **Special Character Detection: DLE, SYNC, SOH, STX, ITB, ETB, ETX**
  - **ASCII or EBCDIC**
  - **Non-Transparent Mode and Transparent Mode**
  - **8-Bit Character Length**
  - **Automatic Fill Character Insertion with Selectable Stripping**
  - **CCITT or CRC-16 Error Detection**
  - **Interrupt on End of Message**
- **Byte Control Protocols: DDCMP and Other Programmable SYNC Characters**
  - **5- to 8-Bit Character Length**
  - **Selectable CRC Error Detection**
  - **Automatic Fill Character Insertion with Selectable Stripping**
- **Directly Addressable Parameter Control Registers: Mode, SYNC/Address, Transmitter Control, and Receiver Control**
- **Separate Addressable Status and Data Registers for Receiver and Transmitter**
- **Modem Handshake Signals: $\overline{RTS}$, $\overline{CTS}$, $\overline{DTR}$, $\overline{DSR}$, and $\overline{CD}$**
- **NRZ or NRZI (Zero-Complementing)**
- **Full- or Half-Duplex Operation**
- **Self-Test Loop Mode**
- **8- or 16-Bit Bidirectional 3-State Data Bus**
- **TTL-Compatible**
- **Single +5 V Supply**
- **40-Pin Package**

Connection Diagram
40-Pin DIP



(Top View)

*F6856 Designation
†F3846 Designation

5

Input/Output Designations

Name	Type	Function
D_0-D_{15}	I/O	Data Bus: D_0-D_{15} contain bidirectional data, status, and control information to and from the CPU. D_0-D_7 may be wired-OR to D_8-D_{15} for use as an 8-bit data bus.
A_0-A_2	I	Register Address: A_0-A_2 select internal data, status, and control registers. The internal registers may be selected as eight or 16 bits. See *Register Address* section.
BYTE	I	Byte: A HIGH level indicates an 8-bit data bus. A LOW level indicates a 16-bit bus.
$\overline{CE}$	I	Chip Enable: A LOW level enables a data bus transfer with E.
*R/$\overline{W}$	I	Read/$\overline{Write}$: A HIGH level allows data from the addressed register to be output to the data bus. A LOW level allows data from the bus to be loaded into the addressed register.
*E	I	Enable: A strobe on this input causes information transfer between the data bus and the addressed register when the $\overline{CE}$ input is LOW.
$\overline{CI}$	I	Chip Initialize: A LOW level initializes the internal control registers and timing.
RCLK	I	Receiver Clock: RCLK provides timing for the receiver logic. RCLK frequency is the same as the received baud rate.
RSI	I	Received Serial Input: RSI is the received serial data. Data changes on the negative going edge of RCLK.
TCLK	I	Transmitter Clock: TCLK provides timing for the transmitter logic. TCLK frequency is the same as the transmitted baud rate.
TSO	O	Transmitter Serial Output: TSO is the transmitted serial data. Data changes on the positive going edge of TCLK.
RDA	O	Receiver Data Available: A HIGH level indicates an assembled character is in the receiver buffer. RDA is reset on the trailing edge of E when the receiver buffer is read by the CPU.
RSOF	O	Received SYNC or FLAG: RSOF is HIGH for one receiver clock period each time a received SYNC or FLAG character is detected.
TBMT	O	Transmitter Buffer Empty: A HIGH level indicates the device is ready to receive new data and/or control information from the CPU. TBMT is reset on the leading edge of the first TCLK following the trailing edge of E when the transmitter buffer is loaded.
$\overline{IRQ}$	O	Interrupt Request: The $\overline{IRQ}$ output goes LOW to indicate a change in the internal status of the device. The status bits linked to the $\overline{IRQ}$ output are receiver overrun (ROVR), received end-of-message (REOM) and transmitter underrun (TUR). $\overline{IRQ}$ is reset on the trailing edge of E when the associated status register is read.
$\overline{DTR}$	O	Data Terminal Ready: The $\overline{DTR}$ output is general-purpose in nature. It can be set LOW by programming the appropriate bit of the receiver control register.
$\overline{DSR}$	I	Data Set Ready: The $\overline{DSR}$ input is general-purpose in nature. It can be tested by the CPU by reading the transmitter status register.
$\overline{CD}$	I	Carrier Detect: The $\overline{CD}$ input is general-purpose in nature. It can be tested by reading the transmitter status register.
$\overline{RTS}$	O	Request to Send: $\overline{RTS}$ is used with $\overline{CTS}$ to enable the transmitter. It may be set LOW by programming the appropriate bit of the transmitter control register.
$\overline{MISC}$	O	Miscellaneous: The $\overline{MISC}$ output is general-purpose in nature. It can be set LOW by programming the appropriate bit of the receiver control register.
$\overline{CTS}$	I	Clear to Send: $\overline{CTS}$ is used with $\overline{RTS}$ to enable the transmitter. It can be tested by reading the transmitter status register.
V_{DD}	I	Power Supply Input: +5 V ±5%.
V_{SS}	I	Ground: 0 V reference.
†$\overline{RD}$	I	Read Pulse: Pulse (negative) on this input with address and $\overline{CE}$ transfers the addressed data register contents to the data bus.
†$\overline{WR}$	I	Write Pulse: Pulse (negative) on this input with address and $\overline{CE}$ transfers the data bus information to the addressed register.

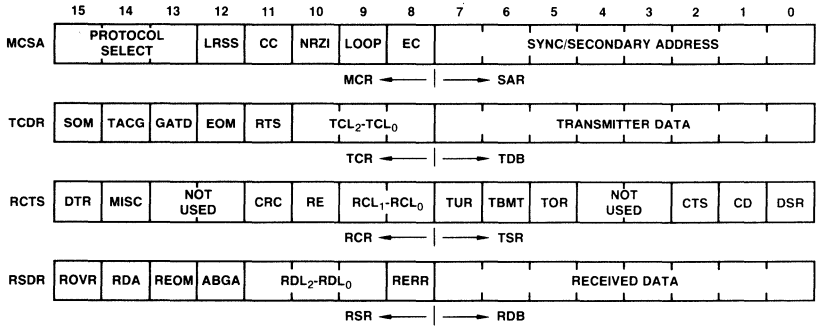
*Pin label for F6856
†Pin label for F3846

Register Definitions

		Bits	Description
Addressable			
MCSA	Mode Control SYNC/Address	16	The upper eight bits (MCR) contain mode control information common to the receiver and transmitter. The lower eight bits (SAR) contain the programmed SYNC character in BCP or the secondary address in BOP. It is not used in BISYNC mode.
TCDR	Transmitter Control and Data Register	16	The upper eight bits (TCR) contain control information specifically for the transmitter. The lower eight bits (TDB) contain the data character to be transmitted.
RCTS	Receiver Control and Transmitter Status Register	16	The upper eight bits (RCR) contain control information specifically for the receiver. The lower eight bits (TSR) contain transmitter and modem status information.
RSDR	Receiver Status and Data Register	16	The upper eight bits (RSR) contain receiver status information. The lower eight bits (RDB) contain the assembled received character.
Internal Receiver			
RIR	Receiver Input Register	8	RIR, RIB and RSPR are used for character assembly and CCR is used to check for received CRC error.
RIB	Receiver Input Buffer	16	
RSPR	Receiver Serial to Parallel Register	8	
CCR	CCR Check Register	16	
Internal Transmitter			
TXR	Transmitter Shift Register	8	TXR is used to convert parallel data from TDB to a serial output. CGR generates the transmitted CRC check sequence.
CGR	CRC Generation Register	8	

5

Short Form Register Format



Error Control

BOP — A frame check sequence (FCS) is transmitted/received as a 16-bit character following the last data character of a frame. The CRC polynomial used to generate/check the FCS is CRC-CCITT ($X^{16} + X^{12} + X^5 + 1$) with the dividend preset to "0"s or "1"s.

BISYNC — A block check character (BCC) is transmitted/received as a 16-bit character following an ITB, ETB or ETX character. The CRC polynomial used to generate/check the BCC is either CRC-16 ($X^{16} + X^{15} + X^2 + 1$) or CRC-CCITT with the dividend preset to "0"s.

BCP — A BCC (for 8-bit characters only) twice the data character length is transmitted/received following the last data character of a message if CRC is selected. The CRC polynomial used to generate/check the CRC is CRC-16 or CRC-CCITT preset to "0"s. No error check is available for character lengths less than eight bits.

Special Characters

Character	Bit Pattern	Function
BOP		Frame
FLAG	01111110	Message
ABORT	11111111 Generated 11111110 Detected	Terminate a message prematurely
GA	11111110	Close frame in store loop mode
ADDRESS	SAR	Secondary station address
BISYNC		
SYNC	00010110 ASCII 00110010 EBCDIC	Start a message and fill character
PAD	11111111	End-of-frame pad
DLE	00010000	Data link escape
SOH	00000001	Start of heading
STX	00000010	Start of text
ITB	00011111	End of intermediate transmission block
ETB	00010111 ASCII 00100110 EBCDIC	End of transmission block
ETX	00000011 ETX	End of transmission
BCP		
SYNC	SAR	Start a message and fill character
PAD	11111111	End-of-frame pad, selectable fill character for DDCMP.

Functional Description

The SPCC is functionally partitioned into receiver, transmitter, addressable registers, and data bus control. *Figure 1* is a block diagram of the SPCC; *Figures 2* and *3* show the data flow in the receiver and transmitter, respectively.

Receiver

The mode control SYNC address (MCSA) register must be programmed prior to starting receiver operation. The receiver may then be enabled and the character length established by programming the receiver control register (RCR). Once the receiver is enabled, data on the RSI input will be serially shifted into the receiver input register (RIR). Data is decoded from NRZI to NRZ as it is continuously monitored, on a bit-for-bit basis, for a match with the FLAG (BOP) or SYNC (BISYNC or BCP) character. The RSOF output is set HIGH for one RCLK clock period when a match occurs. The receiver then operates as described below for each mode of operation.

BOP Operation — A flow chart of BOP receiver operation is shown in *Figure 4*. The receiver starts assembling characters and accumulating the CRC immediately after the detection of a FLAG. It also continues to search for additional FLAG, ABORT or GA characters on a bit-for-bit basis. Zero deletion (to remove "0"s added to the data stream after five consecutive "1"s to distinguish data from FLAG, ABORT and GA) is implemented in the RIR after the FLAG detection logic.

Assembled characters are shifted through the receiver input buffer (RIB) into the receiver serial-to-parallel register (RSPR) and transferred to the receiver data buffer (RDB). The RDA output and status bit are set HIGH each time data is transferred to RDB. Receiver data should be read by the CPU before the next character is assembled to prevent an overrun, resulting in loss of data. The $\overline{IRQ}$ output will go LOW and the ROVR status bit will be set if an overrun occurs.

Character length assembly is set at eight bits per character at the start of each frame. It remains at eight bits until the address and control fields have been processed. (See *Figure 5*). Character length switches to the programmed length at the start of the information field, if any, until the closing FLAG, ABORT or GA is detected. The length of the address field is determined by monitoring the least significant bit (LSB) of each address character for a logic "1". The last character of the address field has a "1" in the LSB. The length of the control field is one or two bytes, as programmed in the MCR.

Fig. 1 Block Diagram

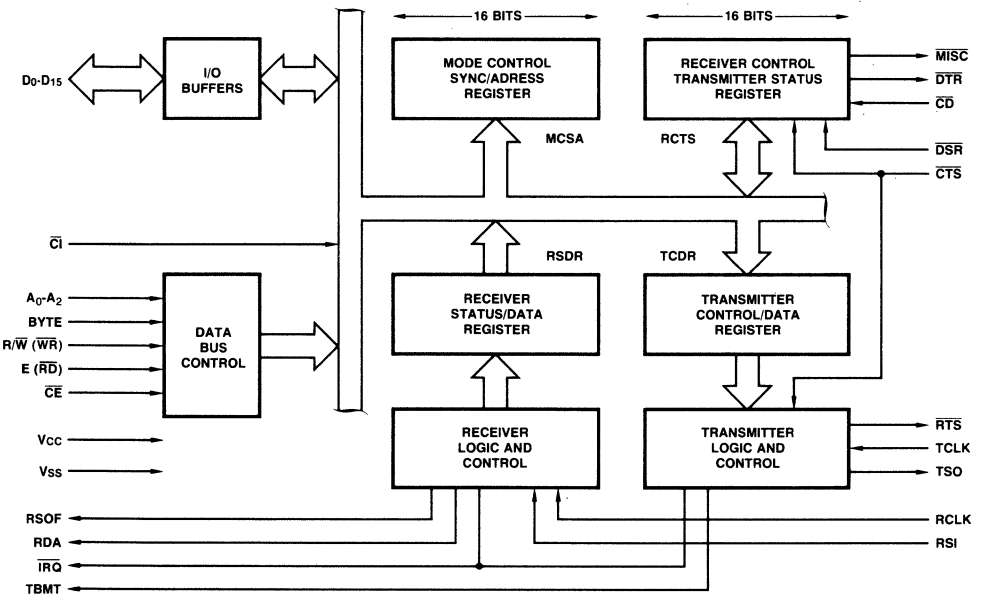


Fig. 2 Receiver Data Path

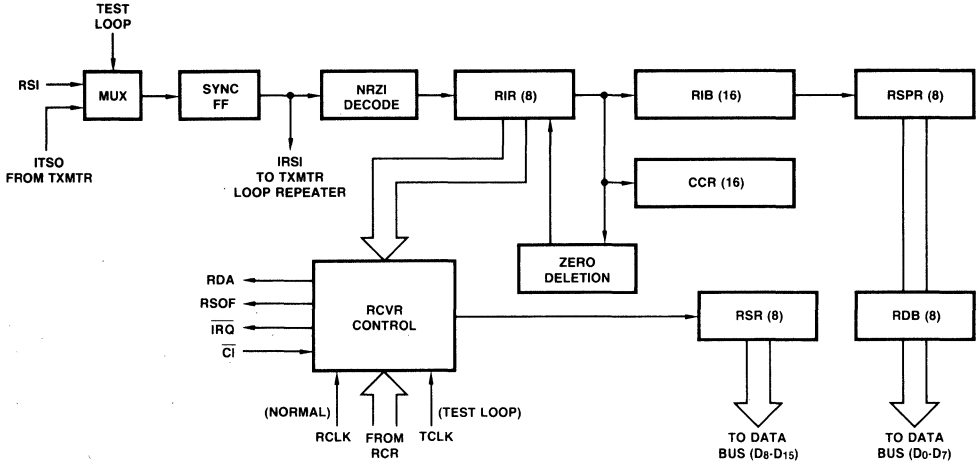


Fig. 3 Transmitter Data Path

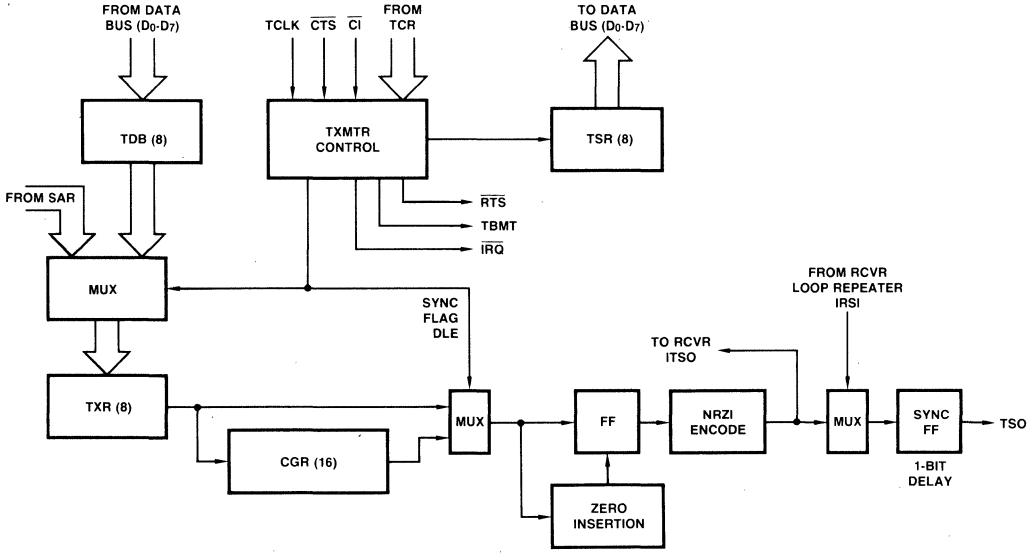


Fig. 4 BOP Receive Flow Chart

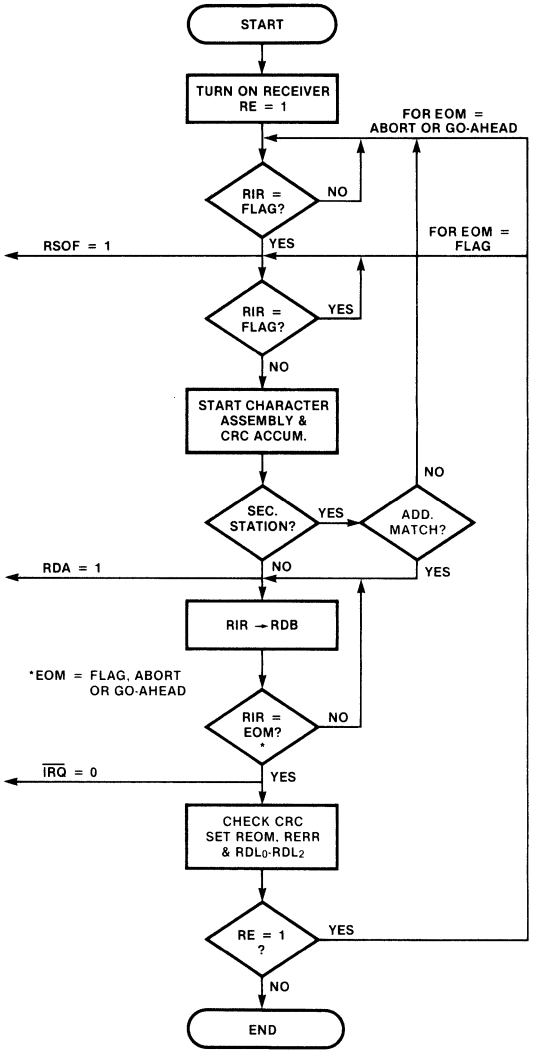
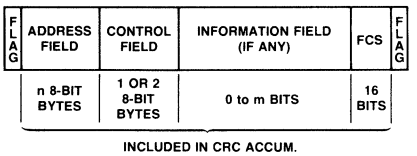


Fig. 5 BOP Message Format



Character assembly and CRC accumulation are stopped when a closing FLAG, ABORT or GA is detected. REOM, ABGA (if the closing character was an ABORT or GA), RDL_0-RDL_2 (indicating length of last character) and RERR (if the accumulated CRC is incorrect) status bits are set. The last character is transferred to RDB, the RDA output is set HIGH and the $\overline{IRQ}$ output is set LOW.

The CRC accumulation includes all characters following the opening FLAG through the frame check sequence (FCS). The contents of the CRC check register (CCR) are checked at the close of a frame if CRC is selected. If an error is detected, RERR status bit is set. Neither the FCS nor the closing FLAG are assembled and passed on to the CPU.

The receiver may be turned off after the status and last characters are read by the CPU by resetting the RE bit of RCR, or it can be left active to receive additional frames.

The closing FLAG of one frame may be used as the opening FLAG of the next frame. Character assembly of the next frame starts with the first non-FLAG character. If the frame was closed with an ABORT or GA, an opening FLAG must be detected before character assembly of the next frame is started.

All receiver status bits except RDA are reset after the receiver status register (RSR) is read by the CPU. The RDA output and status bit are reset when RDB is read by the CPU.

If secondary address is selected, the first non-FLAG character of a frame is compared to the contents of the SYNC/Address Register (SAR). Data for the frame is not passed on to the CPU if no address match occurs. When GLOBAL address is selected, an all '1s' address also results in address match.

Loop Repeater Operation — Loop repeater mode is a special case of BOP. Receiver operation is the same as for BOP, except that the NRZI decode logic is disabled, frames may be terminated by a GO-AHEAD or FLAG, and received data and GA are routed to the transmitter. The RCLK and TCLK lines should be tied together in this mode.

BISYNC Operation — A flow chart of BISYNC receiver operation is shown in *Figure 6,* and the BISYNC message format is illustrated in *Figure 7.* Characters in BISYNC mode may be either EBCDIC or ASCII, as programmed in the MCR. Character length defaults to eight bits. The eighth bit, when ASCII is programmed, may be used for odd parity by the CPU. It is ignored in the recognition of the ASCII characters.

Character assembly starts after receipt of two continuous SYNC characters and continues until the receiver is turned off by resetting the RE bit of RCR. Assembled characters are shifted through the RIB to the RSPR and transferred to the RDB. The RDA output and status bits are set HIGH each time a character is transferred to the RDB. All characters that match the SYNC character in non-transparent mode and DLE SYNC pairs (if not immediately preceded by an odd number of DLEs) in transparent mode are excluded from the RDB. However, the RSOF output goes HIGH for one RCLK clock period each time a SYNC character is detected.

Data must be read by the CPU each time the RDA output goes HIGH before the next character is assembled to prevent an overrun, resulting in loss of data. The $\overline{IRQ}$ output goes LOW and the ROVR status bit is set if an overrun occurs.

The receiver always starts operation in the non-transparent mode. It switches to transparent mode if a DLE STX character pair is received. The receiver will then remain in transparent mode until a DLE ITB, DLE ETB or DLE ETX (if not immediately preceded by an odd number of DLEs) character pair is received.

CRC accumulation begins after the first non-SYNC character if the first character is an SOH or STX. It begins after the second non-SYNC character and enters transparent mode if the first two non-SYNC characters are DLE STX. SYNC characters in non-transparent mode or DLE SYNC pairs in transparent mode are excluded from the CRC accumulation. The first DLE of a DLE DLE sequence and the DLE of DLE ITB, DLE ETB or DLE ETX sequences are not included in the accumulation. The CRC is checked for 0000 remainder after receipt of an ITB, ETB or ETX in non-transparent mode or DLE ITB, DLE ETB or DLE ETX in transparent mode. The REOM and RERR (a non-zero remainder is detected) status bits are set when the closing character is transferred to the RDB, RDA is set HIGH and $\overline{IRQ}$ is set LOW. The block check character (BCC) following the closing character is passed to the CPU as the next two characters. If the closing character was an ETB or ETX, the receiver should be reset by dropping the RE bit of RCR. If the closing character was an ITB, CRC accumulation and character assembly will start again on the first character following the BCC.

All receiver status bits except RDA are reset each time RSR is read by the CPU. The RDA output and status bit are reset each time RDB is read by the CPU.

Fig. 6a BISYNC Receive

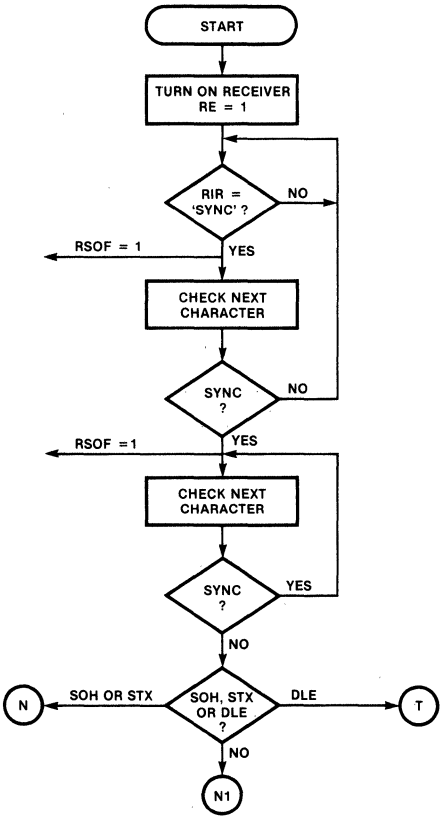


Fig. 6b BISYNC Receive

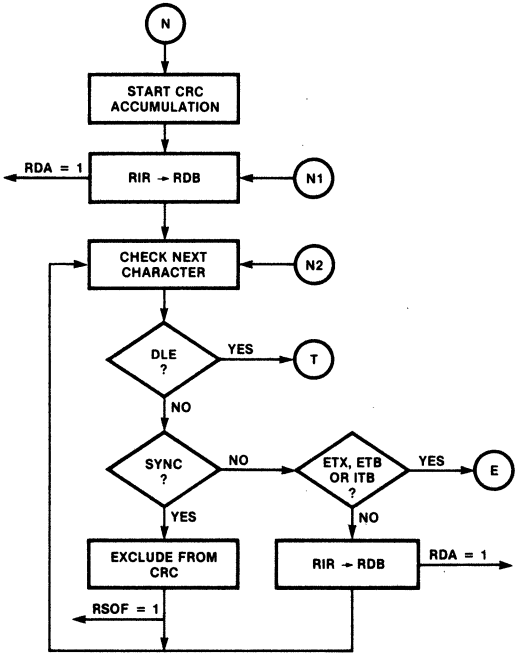
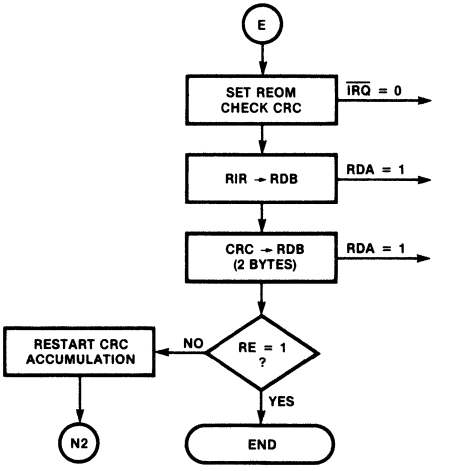


Fig. 6c BISYNC Receive



5

Fig. 6d BISYNC Receive

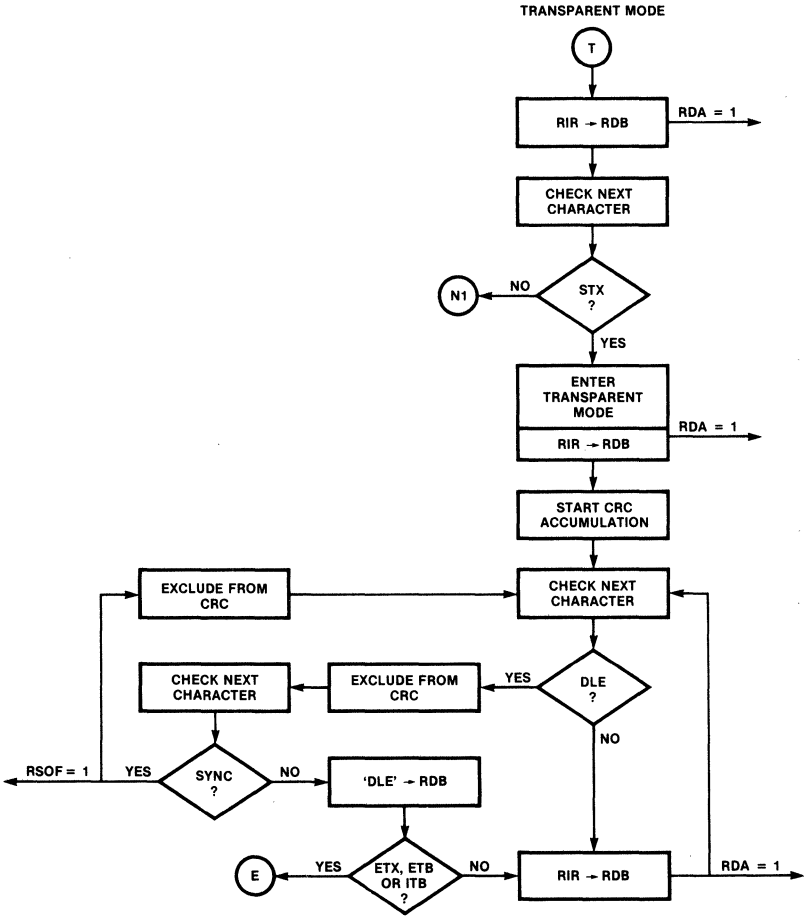
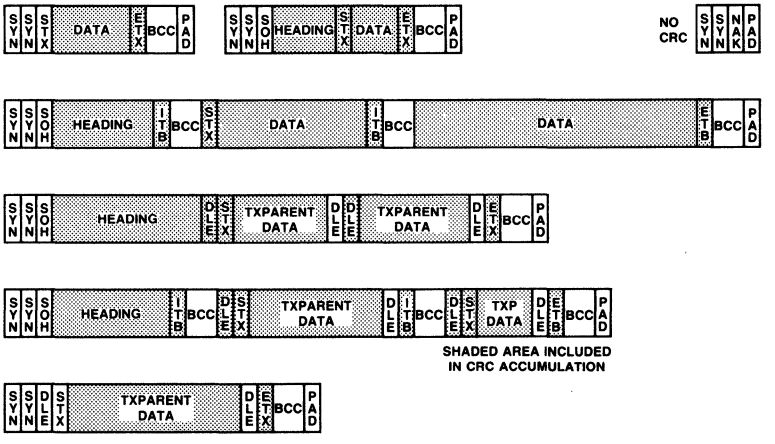


Fig. 7 BISYNC Message Format



Shaded area included in CRC accumulation.

BCP Operation — The flow diagram for BCP mode other than BISYNC is shown in *Figure 8*, and the BCP message format is illustrated in *Figure 9*. The SYNC character is programmed in the SAR. All characters, including the SYNC character are the length specified in the receiver control register (RCR).

Character assembly starts after receipt of two contiguous SYNC characters and continues until the receiver is turned off by resetting the RE bit of RCR. Assembled characters are shifted through the RIB to the RSPR and transferred to the RDB. The RDA output and status bit are set HIGH each time an assembled character is transferred to the RDB. All characters that match the SYNC character are excluded from the RDB, if SYNC strip has been programmed. Only leading SYNC characters are excluded from the RDB if SYNC stripping has not been programmed. However, the RSOF output goes HIGH for one RCLK clock period each time a SYNC character is detected.

Data must be read by the CPU each time the RDA output goes HIGH before the next character is assembled. If not, an overrun will occur resulting in loss of data. The $\overline{IRQ}$ output goes LOW and the ROVR status bit is set if an overrun occurs.

CRC accumulation begins with the first non-SYNC character and includes all subsequent characters if SYNC strip is not programmed. The CRC accumulation will include only non-SYNC characters if SYNC strip is programmed. The CRC accumulation is checked each character time and the RERR status bit is set if the remainder does not equal "0" or reset if the remainder equals "0". Since there is no defined end-of-message (EOM) character, the REOM status bit is not set. The CPU must determine when the end of message occurs and check the RERR status at that time. If an error-free message has been received, RERR will be "0" for one character time. RE may be dropped, thereby resetting the receiver, after the last character has been read. If RE is not reset, CRC accumulation and character assembly will begin again on the first character following the BCC. The two characters of the BCC are output as normal data characters.

Fig. 8a BCP Receive

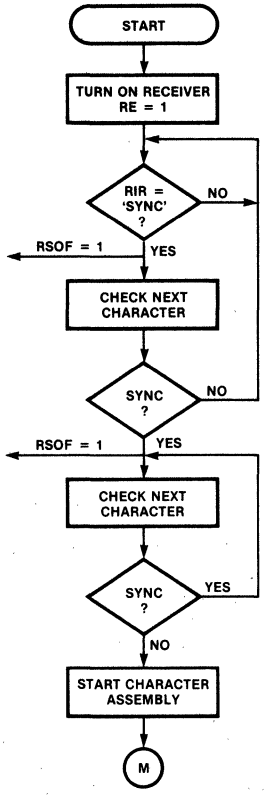


Fig. 8b BCP Receive

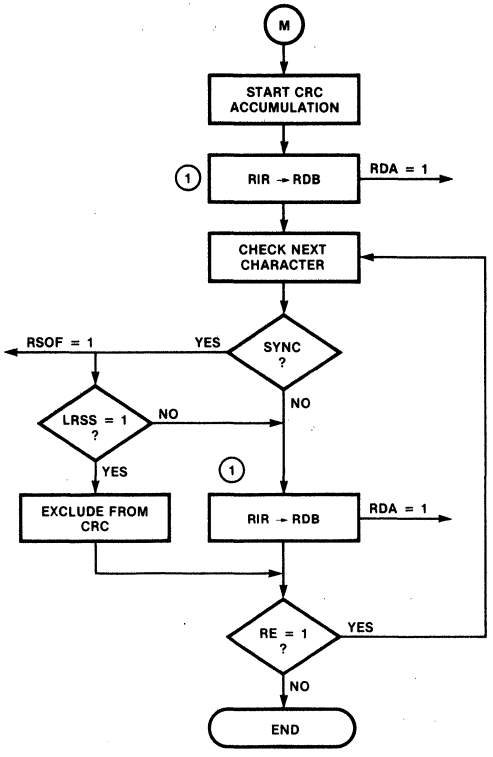
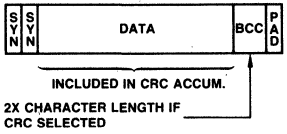


Fig. 9 BCP Message Format



Transmitter

The mode control SYNC/address (MCSA) register must be programmed prior to starting transmitter operation. The CRC bit of the receiver control register must be set if ORC error checking is desired. The RTS bit of the transmitter control register (TCR) must be set to turn on the transmitter. The SOM bit of TCR may also be set at this time and the transmitter data buffer (TDB) loaded with the first character of the message. When RTS has been loaded into TCR, the $\overline{RTS}$ output goes LOW. The TSO output is held HIGH (marks) until the $\overline{CTS}$ input goes LOW. Two SYNC or FLAG characters are then output on TSO, if SOM has been set. Otherwise, TSO will continue to output marks until SOM is set and the first character is loaded into TDB. Transmitter operation after the two SYNC or FLAG characters have been output depends on the mode of operation. Note that TRS and transmitter character length must be reloaded each time TCR is updated until after the EOM bit has been set.

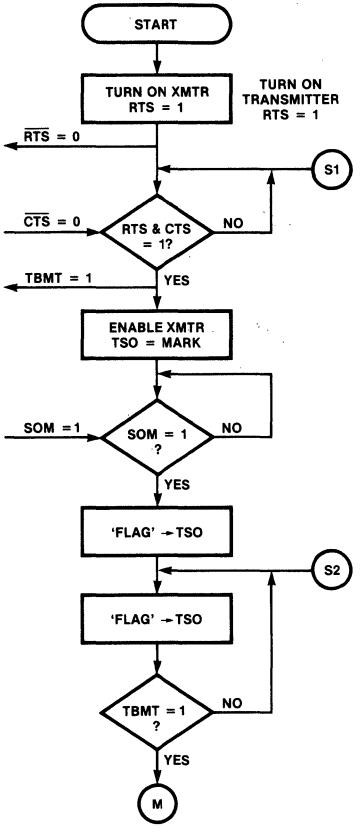
BOP Operation — Character length in BOP mode always starts at eight bits per character each frame. It remains eight bits until the address and control fields have been transmitted. It then switches to the programmed length at the start of the information field, if any, until the last character has been transmitted. Character length switches back to eight bits for the transmission of the frame check sequence (FCS) and the closing FLAG.

A flow diagram for BOP transmitter operation is shown in *Figure 10.* The secondary address is transmitted after the initial two FLAGs. The secondary address comes from the SYNC/address register (SAR) if the device is programmed as a secondary station or from the TDB if the device is programmed as a primary. If the secondary address came from SAR, it is followed in the transmission by the character from TDB. Characters are transferred in parallel from SAR or TDB to the transmitter shift register (TXR) and serially shifted, LSB first, out the TSO output. The TBMT output and status bit are set HIGH each time data is transferred from TDB. The CPU must update TCR, if required, and load TDB with the next character. An underrun occurs if this is not done within one character time. If an underrun occurs, the TUR status bit is set and an ABORT (11111111) is transmitted. The output is held at a mark until SOM is set for a new message. A transmitter overrun occurs if TDB is updated before TBMT goes HIGH. An overrun can result in the misinterpretation or loss of the character in TDB. The TOR status bit is set when an overrun occurs.

The least significant bit (LSB) of each character, starting with the secondary address is examined. The first character with an LSB = "1" denotes the last character of the address field. The next one or two characters (programmed in MCR) are the control field. The character length switches to the programmed length in TCR after the last character of the control field, unless that character was the end of message.

The CPU must set the EOM bit of TCR when loading the last character of the message. Character length may be changed at this time to allow transmission of a residual last character. The character in TDB is followed by the FCS (if CRC is selected) and a closing FLAG when EOM is set. The transmitter may be turned off by resetting $\overline{RTS}$ after TBMT goes HIGH or it may remain active. The closing FLAG of one frame may be used as the opening FLAG of the next frame by setting SOM and loading TDB after TBMT goes HIGH. If the transmitter is left active and SOM has not been set, FLAG characters are transmitted between frames if the GATD bit of TCR equals "0" or marks if GATD equals "1".

Fig. 10a BOP Transmit

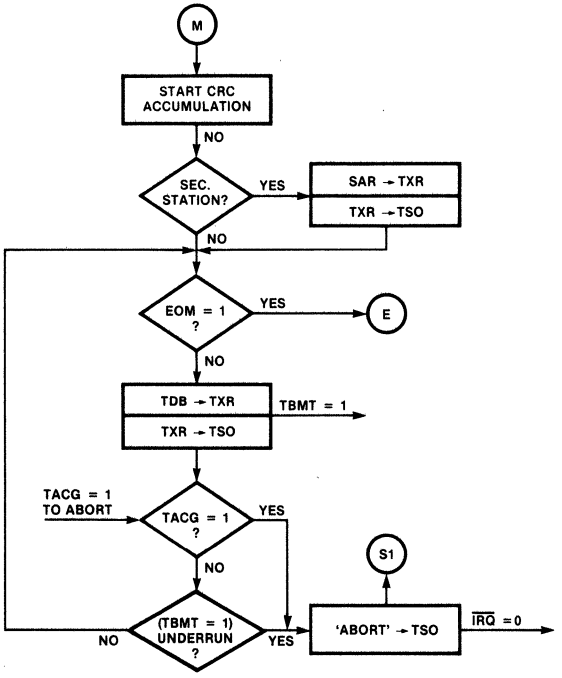


5

A message may be terminated at any time with an ABORT by setting the TACG bit of TCR. This causes the TSO output to go immediately to mark condition until SOM is set.

Data transmitted on the TSO output is monitored continuously for five consecutive "1s." A "0" is inserted in the data stream each time this condition occurs. This insures that a data character will not be interpreted as a FLAG, ABORT or GA at the received end.

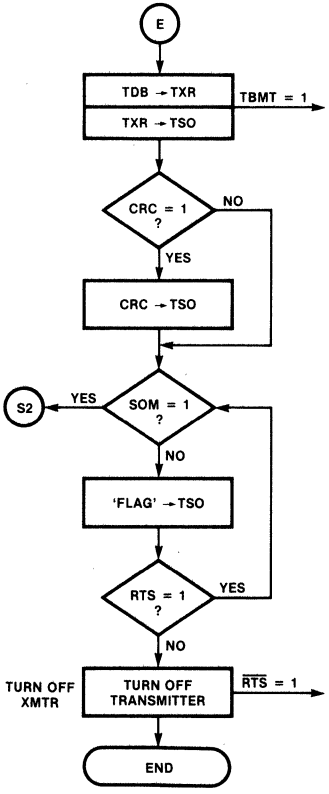
Fig. 10b BOP Transmit



TUR and TOR status bits are reset whenever the transmitter status register (TSR) is read. The TBMT output and status bit are reset when TDB is loaded.

CRC accumulation begins with the first non-FLAG character and includes all subsequent characters up to and including the last data character. The accumulated CRC is then transmitted as the FCS following the last data character, if CRC is selected.

Fig. 10c BOP Transmit



Loop Repeater Operation — Loop repeater mode is a special case of BOP. The primary station in the loop should be programmed for normal BOP primary operation. The GATD bit of TCR is used to initiate a polling sequence. When this bit is set, marks are transmitted after the closing FLAG of a frame. The last "0" of the closing FLAG and the next seven "1s" are interpreted down-loop as a GO-AHEAD. The end of the polling sequence is detected when the ABGA (received GA) bit of the RSR is set.

Down-loop stations should be programmed as BOP secondary, loop repeater (LRSS = "1" in MCR). In this mode, data received at the RSI input is delayed one bit time and output on TSO. When data is to be transmitted in this mode, the CPU should set $\overline{RTS}$ and SOM and load the first character into TDB. The $\overline{CTS}$ input is ignored in this mode. The transmitter waits for a received GA. When a received GA is detected, the seventh "1" is changed to a "0," creating a FLAG. This prevents the down-loop station from receiving a GA, reserving the line for the transmitting station. The TBMT output and status bit are set and transmitter operation proceeds in normal BOP operation, except that the NRZI encode logic is disabled.

When the last character and FCS have been transmitted, the message is terminated with a GA. The TSO output switches back to RSI delayed one bit time. Down-loop stations may then capture the line by detecting the GA.

The RCLK and TCLK lines should be tied together in this mode.

BISYNC Operation — A flow diagram for BISYNC transmitter operation is shown in *Figure 11.* Character length for BISYNC mode defaults to eight bits per character. The transmitter always assumes non-transparent mode unless forced to transparent mode by the CPU.

The message format following the initial SYNC pair depends on the action of the CPU. If the transmitter data buffer (TDB) has not been loaded with the first character of the message, SYNC characters are output on TSO until a TDB load. This can occur only with an 8-bit data bus, since TCR and TDB are loaded simultaneously for a 16-bit data bus. The character from TDB, when available, is transferred to the transmitter shift register, (TSR) and serially shifted out the TSO output. The character in TDB is preceded by a contiguous DLE when GATD (transmit DLE) is set. The GATD bit is cancelled after it has been internally processed. The first occurrence is interpreted as a DLE STX command and the transmitter begins transparent mode operation. The transmitter will remain in transparent mode until the end of the message.

The TBMT output and status bit are set HIGH each time data is transferred from TDB. The CPU should update TCR, if required, and load TDB with the next character. An underrun occurs if this is not done within one character time, and the TUR status bit is set and SYNC characters (or DLE SYNC pairs in transparent mode) are transmitted until TDB is updated. A transmitter overrun occurs if TDB is updated before TBMT goes HIGH. An overrun can result in the misinterpretation or loss of the character in TDB. The TOR status bit is set when an overrun occurs.

The EOM bit of TCR, GATD (if in transparent mode) and TACG (if the accumulated CRC is to be transmitted as the block check character) should be set when the last character is loaded into TDB. The last character must be an ITB, ETB or ETX if CRC is used. A 16-bit BCC, if selected, is transmitted following the last character. The last character is followed by marks for a minimum of one character time if no BCC is transmitted.

A second block of data may be transmitted immediately following the BCC by setting SOM and loading TDB after TBMT goes HIGH. The transmitter may be turned off at this time by resetting RTS. The transmitter transmits marks following the BCC for a minimum of one character time if SOM is not set.

CRC accumulation begins after the first non-SYNC character for non-transparent mode, or after the second non-SYNC character if the message starts in transparent mode. The CRC continues up to and including the last character. SYNC characters or DLE SYNC pairs caused by a transmitter underrun are not included. Forced DLE characters in transparent mode are not included. The forced DLE of a DLE STX pair which occurs after the start of the message is included. (See *Figure 7.)*

TUR and TOR status bits are reset whenever the transmitter status register (TSR) is read. The TBMT output and status bit are reset when TDB is loaded.

Fig. 11a BISYNC Transmit

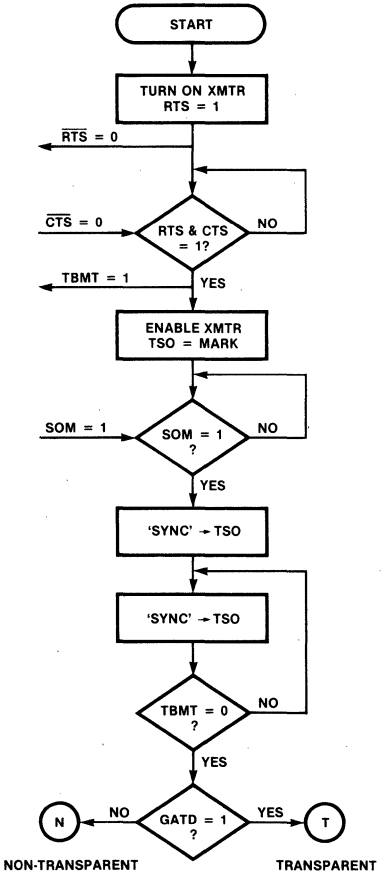
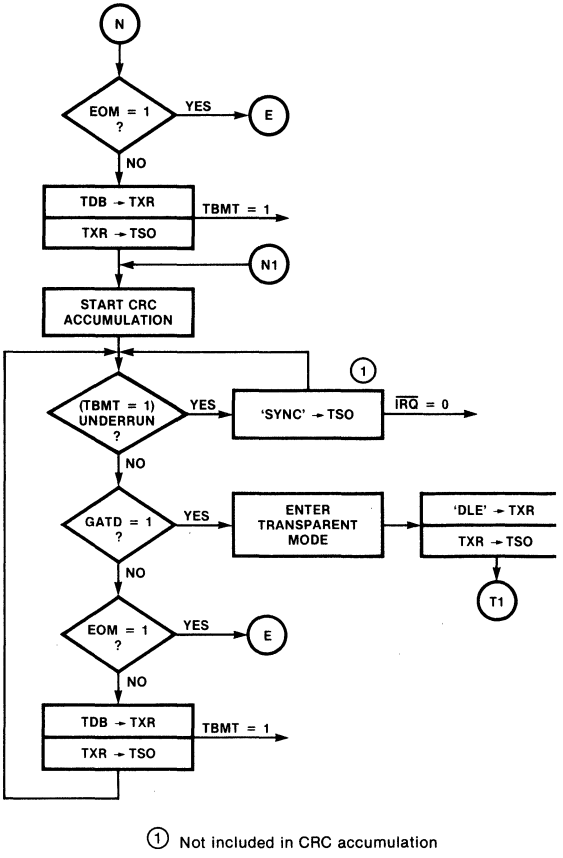
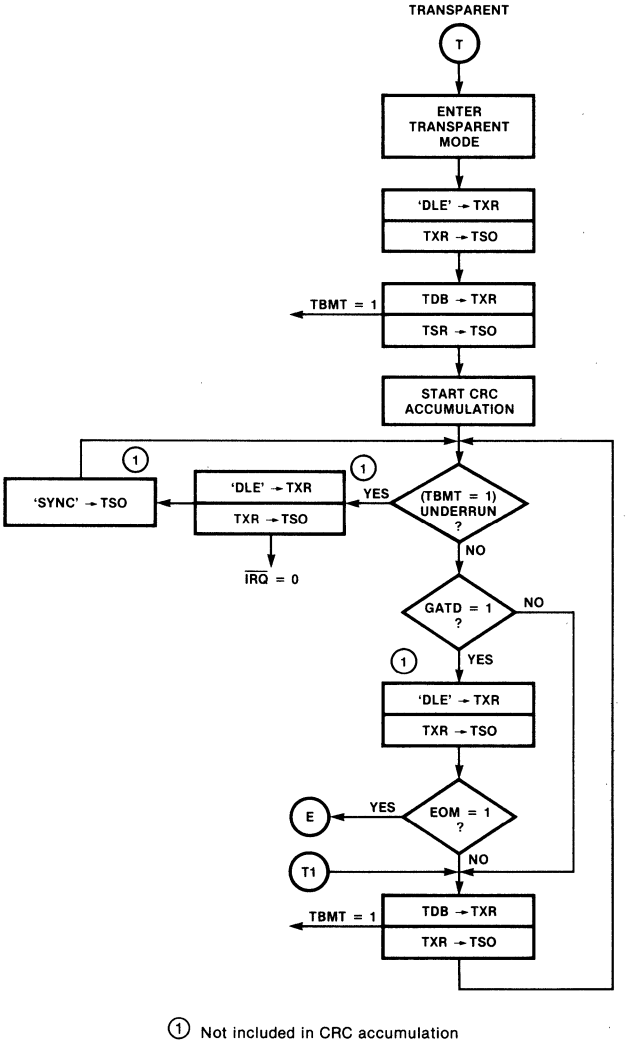


Fig. 11b BISYNC Transmit



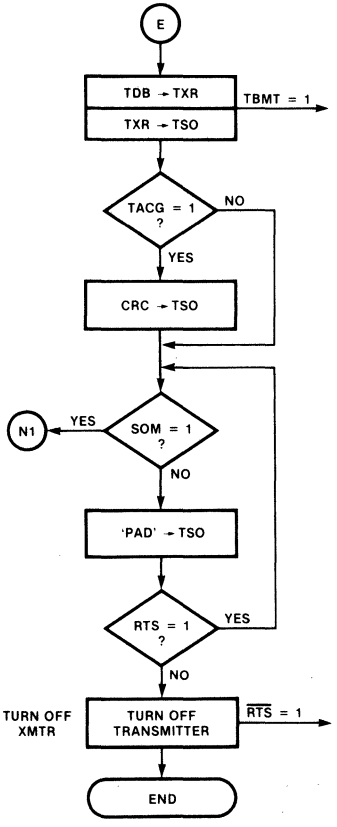
① Not included in CRC accumulation

Fig. 11c BISYNC Transmit



① Not included in CRC accumulation

Fig. 11d BISYNC Transmit



BCP Operation — The flow diagram for BCP mode other than BISYNC is shown in *Figure 12*. The SYNC character is programmed in the SYNC/address register (SAR). All characters are the length specified in the transmitter control register (TCR).

The message format following the initial SYNC pair depends on the action of the CPU. If the transmitter data buffer has not been loaded with the first character of the message, SYNC characters are transmitted until a TDB load. This can occur only with an 8-bit data bus, since TCR and TDB are loaded simultaneously for a 16-bit data bus. The character from TDB, when available, is transferred to the Transmitter Shift Register (TXR) and serially shifted out the TSO output. The TBMT output and status bit are set HIGH each time data is transferred from TDB. The CPU should update TCR, if required, and load TDB with the next character. An underrun occurs if this is not done within one character time, and the TUR status bit is set and SYNC characters (marks, if SYNC stripping is not programmed) are transmitted until TDB is updated. A transmitter overrun occurs if TDB is updated before TBMT goes HIGH. An overrun can result in the misinterpretation or loss of the character in TDB. The TOR status bit is set when an overrun occurs.

The EOM bit of TCR and TACG (if the accumulated CRC is to be transmitted as the block check character) should be set when the last character is loaded into TDB. The last character is followed by a BCC and a pad character if CRC is selected, or the pad character only if CRC is not selected. The transmitter may be turned off by resetting RTS after TBMT goes HIGH.

CRC accumulation (see *Error Control* table) includes all non-SYNC characters. The CRC generation register (CGR) in BCP mode is defined as twice the character length.

TUR and TOR status bits are reset whenever the transmitter status register (TSR) is read. The TBMT output and status bit are reset when TDB is loaded.

Fig. 12a BCP Transmit

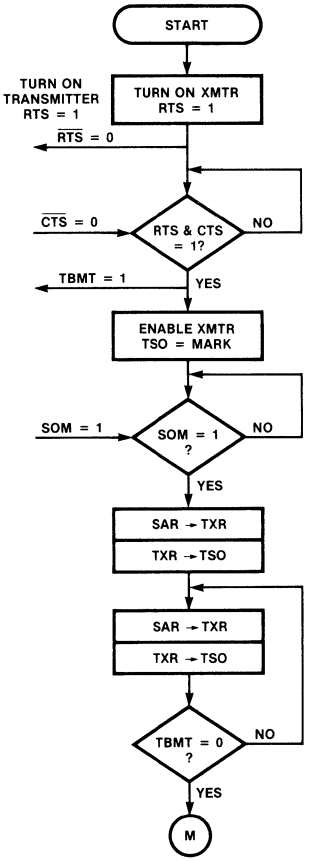
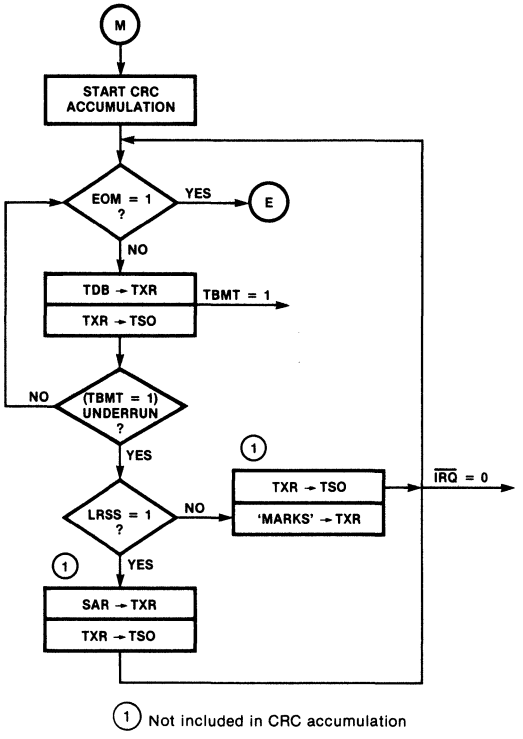
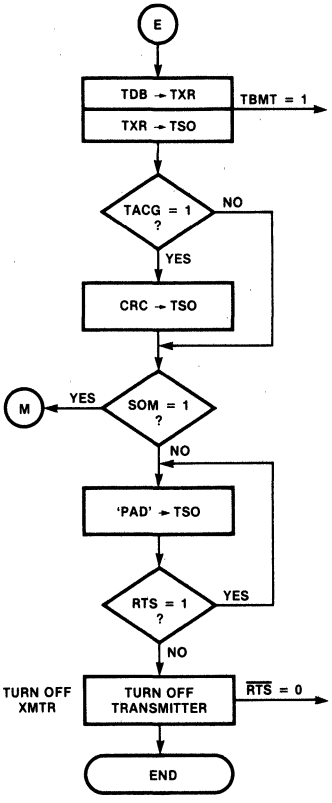


Fig. 12b BCP Transmit



(1) Not included in CRC accumulation

Fig. 12c BCP Transmit



Data Bus Control (F6856)
The CPU uses the register address (A_0-A_2), byte select (BYTE), chip enable ($\overline{CE}$), read/write (R/$\overline{W}$), and enable (E) inputs to control information transfer on the data bus. The byte select input specifies a 16-bit data bus when BYTE = "0" or an 8-bit data bus when BYTE = "1." For an 8-bit data bus, D_0 through D_7 may be wired-OR with the corresponding pins, D_8 through D_{15}.

A read operation (R/$\overline{W}$ = "1") is initiated on the leading edge of E. The other control inputs (A_0-A_2, BYTE, $\overline{CE}$, and R/$\overline{W}$) must be stable before the leading edge of E (see *Bus Timing Characteristics).* Any unused bits in the addressed register are "0." D_8-D_{15} contain receiver status when TSR is read using a 16-bit bus. Status bits are reset on the trailing edge of E when the appropriate register is read.

Data is loaded into the addressed register on the trailing edge of E for a write (R/$\overline{W}$ = "0") operation. The other control inputs must be stable prior to the leading edge of E. TBMT is reset on the trailing edge of E when TCDR (16-bit bus) or TDB (8-bit bus) is addressed.

Data Bus Control (F3846)
Bus Control for the F3846 has the same characteristics as the F6856 with only $\overline{RD}$ required for read rather than both E and R/$\overline{W}$ being "1"s, and only $\overline{WR}$ required for write rather than E being a "1" and R/$\overline{W}$ being a "0."

Register Addresses

	R/$\overline{W}$	A_0	A_1	A_2	Register	$\overline{RD}$	$\overline{WR}$
BYTE = "0"	1	X	0	0	RSDR	0	1
16-Bit	0	X	1	0	TCDR	1	0
Data Bus	0	X	0	1	MCSA	1	0
	1	X	1	1	$RCTS_L$ (TSR)	0	1
	0	X	1	1	$RCTS_U$ (RCR)	1	0
BYTE = 1	1	0	0	0	$RSDR_L$ (RDB)	0	1
8-Bit Data	1	1	0	0	$RSDR_U$ (RSR)	0	1
Bus D_0-D_7	0	0	1	0	$TCDR_L$ (TDB)	1	0
Wired-OR	0	1	1	0	$TCDR_U$ (TCR)	1	0
to D_8-D_{15}	0	0	0	1	$MCSA_L$ (SAR)	1	0
	0	1	0	1	$MCSA_U$ (MCR)	1	0
	1	0	1	1	$RCTS_L$ (TSR)	0	1
	0	1	1	1	$RCTS_U$ (RCR)	1	0

Programming
The mode control SYNC/address (MCSA) register is a directly addressable write-only register used to configure the SPCC for the user's specific data communications environment. MCSA should be programmed after initialization and prior to initiating data transmission or reception. It may be changed at any time that both the receiver and transmitter are disabled. The default mode (after initialization) is BOP primary with one byte control field, NRZI encoding, 8-bit character length, and error control using CRC-CCITT preset to "1s." The lower byte, SYNC/address, is not used in BOP primary mode.

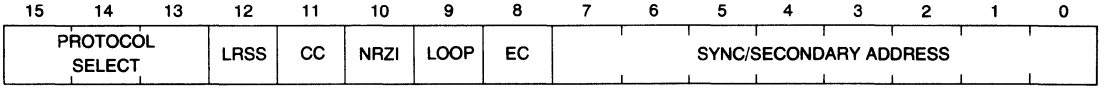
The transmitter control and data register (TCDR) is a directly addressable write-only register that controls the format of the transmitted data. The lower byte (TDB) contains the data characters to be transmitted. The upper byte (TCR) contains control information relating specifically to the data being transmitted. TCDR may be updated whenever the TBMT output is HIGH. The default mode for this register is all "0s" corresponding to transmitter disabled.

The upper byte (RCR) of the receiver control and transmitter status register (RCTS) is a directly addressable write-only register that contains control information specifically related to the receipt of data and the $\overline{\text{DTR}}$ and $\overline{\text{MISC}}$ general-purpose outputs. Those bits that control the received character length should not be changed while the receiver is enabled. The default value of RCR is all "0s", corresponding to receiver disabled and general-purpose outputs at a HIGH level.

Specific definition of the format of the addressable registers is given in the following section. Address information is given in the *Data Bus Control* section.

Addressable Register Format

Mode Control SYNC/Address (MCSA) Register - Write-Only

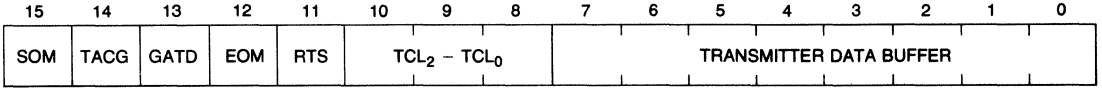


Bit	Name	Mode	Function
0-7	SAR		SYNC/address register
		BOP	Secondary address for secondary station mode
		BISYNC	Not used
		BCP	SYNC character
8	EC		Error control
		BOP	0 = CCITT preset to all "0"s
			1 = CCITT preset to all "1"s
		BISYNC	0 = CRC-16 preset to all "0"s
			1 = CCITT preset to all "0"s
		BCP	Same as BISYNC for 8-bit character length only
9	LOOP	All	Self-test loop mode, TSO loop to RSI internally
10	NRZI	All	0 = NRZ data
			1 = NRZI, zero complementing
11	CC	BOP	0 = 1 control byte, 1 = 2 control bytes
		BISYNC	Not used
		BCP	Not used
12	LRSS		Loop repeater/SYNC strip
		BOP	0 = Normal mode
			1 = Loop repeater mode
		BISYNC	Not used
		BCP	0 = Tx mark for FILL character (strip leading SYNCs only)
			1 = Tx SYNC for FILL character (strip all SYNCs)
13-15		All	Protocol select 15 14 13
			0 0 0 BOP, Primary
			0 1 0 BOP, Secondary
			0 1 1 BOP, Secondary, Global
			1 0 0 BCP
			1 1 0 BISYNC - ASCII
			1 1 1 BISYNC - EBCDIC
			0 0 1 Reserved
			1 0 1 Reserved

5

Addressable Register Format (Cont'd)

Transmitter Control and Data Register (TCDR) - Write-Only



Bit	Name	Mode	Function
0-7	TDB	All	Transmitter data buffer
8-10	TCL_0-TCL_2		Transmitter character length
		BOP/BCP	8 9 10 0 0 0 8 bits 1 0 0 1 0 1 0 2 1 1 0 3 0 0 1 4 1 0 1 5 0 1 1 6 1 1 1 7
		BISYNC	Character length automatically 8 bits
11	RTS	All	Request to Send. "0" = "1" on $\overline{RTS}$ output; "1" = "0" on $\overline{RTS}$ output.
12	EOM	All	End of message. "1" defines character in TBD as last data character of message. This bit is self-cancelling.
13	GATD		Go-ahead/transmit DLE
		BOP	"0" = FLAGs transmitted between frames "1" = Marks transmitted between frames
		BISYNC	"1" = Transmit DLE character ahead of character in TDB. Enter transparent mode.
		BCP	Not used
14	TACG		Transmit abort/CRC generate
		BOP	"1" = Transmit abort
		BISYNC/ BCP	"0" = No CRC on transmitted message "1" = Transmit block check character after last data character
15	SOM	All	Start of message. Initiates start of message, causing SYNCs or FLAGs to be transmitted. This bit is self-cancelling.

Addressable Register Format (Cont'd)

Receiver Control and Transmitter Status Register (RCTS) - Read/Write

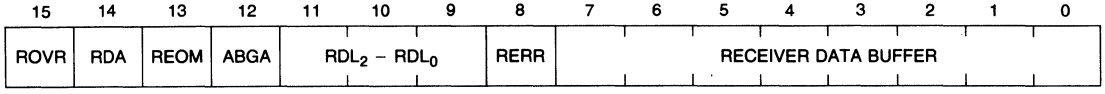
15	14	13 12	11	10	9 8	7	6	5	4 3	2	1	0
DTR	MISC	NOT USED	CRC	RE	RCL_1 – RCL_0	TUR	TBMT	TOR	NOT USED	CTS	CD	DSR

Bit	Name	Mode	Function
0	DSR	All	Data set ready; equals "1" when $\overline{DSR}$ input is LOW.
1	CD	All	Carrier detect; equals "1" when $\overline{CD}$ input is LOW.
2	CTS	All	Clear to send; equals "1" when $\overline{CTS}$ input is LOW.
3-4			Not used
5	TOR	All	Transmitter overrun; "1" = CPU updated TCDR before the SPCC was ready.
6	TBMT	All	Transmitter buffer empty; "1" = CPU may load new data and/or control information in TCDR.
7	TUR	All	Transmitter underrun; "1" = CPU failed to load TDB in time. Abort is transmitted in BOP mode. When TUR occurs, FILL characters are transmitted in BISYNC or BCP. TUR occurs along with a LOW level of $\overline{IRQ}$ output.
8-9	RCL_0-RCL_1	All	Receiver character length 8 9 0 0 8-bits 1 0 5 0 1 6 1 1 7
10	RE	All	Receiver enable; "1" enables receiver
11	CRC	All	"0" = No CRC (Transmit/Receive) "1" = CRC selected
12-13			Not used
14	MISC	All	Miscellaneous; "0" = "1" on $\overline{MISC}$ output; "1" = "0" on $\overline{MISC}$ output.
15	DTR	All	Data terminal ready; "0" = "1" on $\overline{DTR}$ output; "1" = "0" on $\overline{DTR}$ output.

5

Addressable Register Format (Cont'd)

Receiver Status and Data Register (RSDR) - Read Only



Bit	Name	Mode	Function
0-7	RDB	All	Receiver data buffer
8	RERR	All	Received error; "1" = CRC error occurred on received message. Asserted when last character is in RDB.
9-11	RDL_0-RDL_2	BOP only	Received last character length; corresponds to the number of bits in last character. 000 = 8 bits, 100 = 1 bit, 010 = 2 bits, etc.
12	ABGA	BOP only	Abort/go-ahead; corresponds to received abort if RERR = "1" or go-ahead if RERR = "0".
13	REOM	BOP BISYNC	Received end-of-message "1" = received FLAG, abort or go-ahead "1" = received ITB, ETB, or ETX (preceded by DLE in transparent mode).
14	RDA	All	Received data available. "1" indicates valid data available in RDB.
15	ROVR	All	Receiver overrun. "1" indicates CPU failed to read data in RDB before next character was assembled. Accompanied by a LOW on $\overline{IRQ}$ output.

Absolute Maximum Ratings

Operating Temperature	
Ceramic	−55 °C, +125 °C
Cermet	−55 °C, +125 °C
Plastic	0 °C, +70 °C
Storage Temperature	−65 °C, +150 °C
Supply Voltage	−0.3 V, +7.0 V
Input/Output Voltage	−0.3 V, +10 V
Input Voltage	−0.3 V, +15 V
Output Voltage	−0.3 V, +10 V

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages.

DC Characteristics Over the Operating Temperature Range

Symbol	Parameter	Min	Typ	Max	Unit	Condition
V_{DD}	Supply Voltage	4.75	5.0	5.25	V	
	Input Voltage					
V_{IL}	Input LOW	−0.3		0.8	V	
V_{ILC}	Clock LOW	−0.3		0.8	V	
V_{IH}	Input HIGH	2.0		V_{DD}	V	
V_{IHC}	Clock HIGH	2.4		V_{DD}	V	
	Output Voltage					
V_{OL}	Output LOW			0.4	V	I_{OL} = 1.6 mA
V_{OH}	Output HIGH	2.4			V	I_{OH} = −300 μA
	Leakage Current					
I_{LI}	Input Leakage			2.5	μA	
I_{LO}	Output Leakage			±10	μA	
I_{DD}	Supply Current			120	mA	V_{DD} = 5.25 V
	Capacitance					
C_I	Input			10	pF	Measured at +27 °C
C_O	Output			15	pF	and 1 MHz
C_{IO}	Bus In			20	pF	

Serial Port Timing Characteristics (Refer to *Figure 13*)

Symbol	Parameter	Min	Max	Unit
t_{RS}	RSI Set-up Time	100		ns
t_{RH}	RSI Hold Time	50		ns
t_{TSD}	Transmit Serial Data		200	ns
t_{CPW}	Clock Pulse Width	400		ns

Bus Timing Characteristics (Refer to *Figure 14*)

Symbol	Parameter		Min	Max	Unit
Read					
t_{cycE}	Enable Cycle Time		1.0		μs
PW_{EH}	Enable Pulse Width, HIGH		450		ns
PW_{EL}	Enable Pulse Width, LOW		430		ns
t_{AS}	Set-up Time, Address and R/$\overline{W}$	F6856	160		ns
	Valid to Enable Positive Transition	F3846	260		ns
t_{DDR}	Data Delay Time			320	ns
t_H	Data Hold Time		10		ns
t_{AH}	Address Hold Time		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input			25	ns
PW_{RL}	$\overline{RD}$, $\overline{CI}$ Pulse Width, LOW		430		ns
Write					
t_{cycE}	Enable Cycle Time		1.0		μs
PW_{EH}	Enable Pulse Width, HIGH		450		ns
PW_{EL}	Enable Pulse Width, LOW		430		ns
t_{AS}	Set-up Time, Address and R/$\overline{W}$	F6856	160		ns
	Valid to Enable Positive Transition	F3846	260		ns
t_{DSW}	Data Set-up Time		195		ns
t_H	Data Hold Time		10		ns
t_{AH}	Address Hold Time		10		ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input			25	ns
PW_{WL}	$\overline{WR}$, $\overline{CI}$ Pulse Width, LOW		430		ns

Fig. 13 Clock and Serial Data

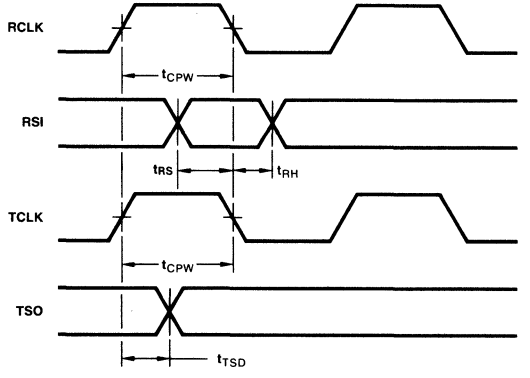
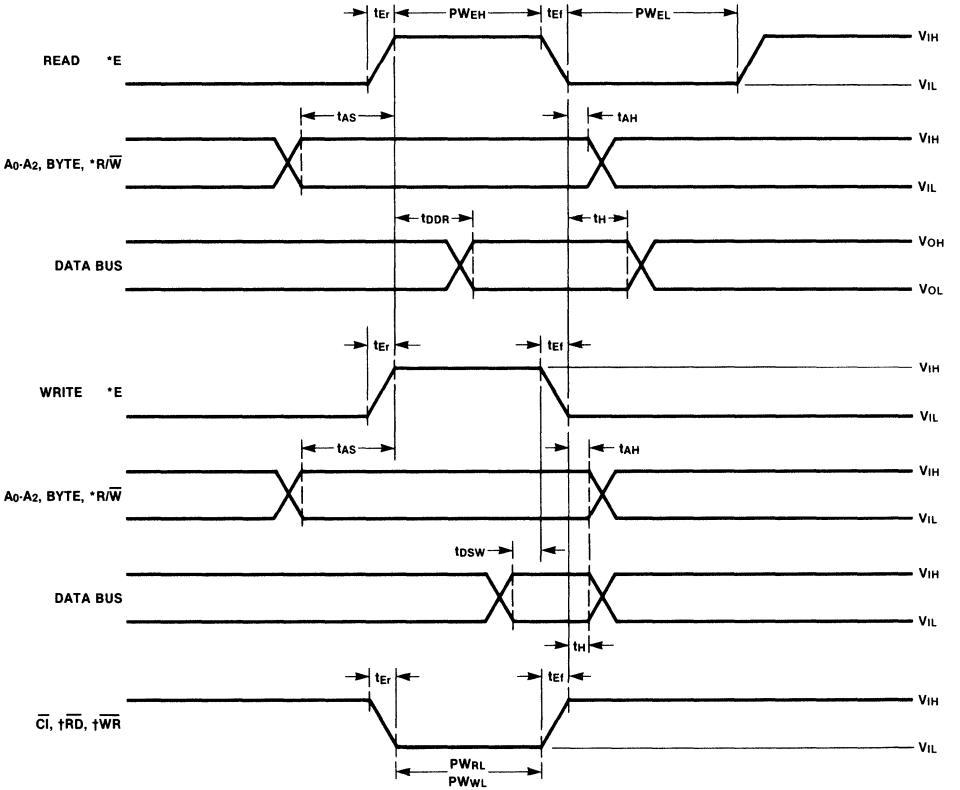


Fig. 14 Read/Write Timing Diagram



*6856
†3846

Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F6856P, S	0 °C to + 70 °C
	F6856CP, CS	− 40 °C to + 85 °C
	F6856DL	− 55 °C to + 85 °C
	F6856DM	− 55 °C to + 125 °C
1.0 MHz	F3846P, S	0 °C to + 70 °C
	F3846CP, CS	− 40 °C to + 85 °C
	F3846DL	− 55 °C to + 85 °C
	F3846DM	− 55 °C to + 125 °C

P = Plastic Package S = CER-DIP Package

FAIRCHILD
A Schlumberger Company

F38456/F68456 Multiple Protocol Communications Controller

Advance Product Information

Microprocessor Product

Description

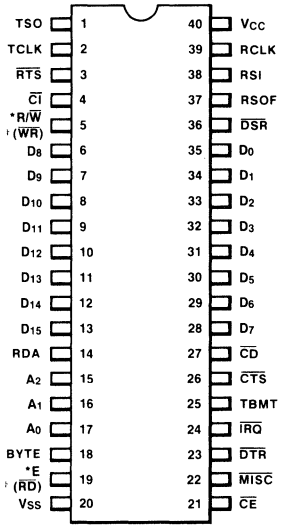
The Fairchild F38456/F68456 Multiple Protocol Communications Controller (MPCC) is a programmable microprocessor peripheral that interfaces a computer system to a serial data communications channel with minimum system overhead. It is designed to satisfy the major interface requirements for asynchronous, synchronous bit-oriented protocols (BOP), or synchronous byte control protocols (BCP). The MPCC is well-suited for application in computer-to-computer communication, or control of network trunk lines.

The MPCC is organized to interface with either an 8- or 16-bit bidirectional data bus, is fully TTL compatible, and operates from a single +5 V supply.

- **F6800, Z80, and 8080 Bus Compatible**
- **Data Rate from DC to 2M BPS**
- **Asynchronous Protocols:**
  - **5- to 8-Bit Character Length**
  - **Parity — Even, Odd or None**
  - **Stops Bits — 1, 1.5 or 2**
  - **Clock Rates — 1X, 8X, 16X or 32X Baud Rate**
  - **Interrupt on Received Parity or Framing Error**
- **Bit-Oriented Line-Control Protocols — SDLC, ADCCP, HDLC**
  - **Automatic Detection and Generation of Special Control Sequences (e.g., Flag, Abort, Go-Ahead)**
  - **Zero Insertion and Deletion**
  - **Primary or Secondary Station Select**
  - **Secondary Station Address Recognition**
  - **Global Address Recognition**
  - **Automatic Extended Address**
  - **One or Two Control Bytes**
  - **Data Character Length from 5- to 8-Bits with 1- to 8-Bit Residual Last Character**
  - **CCITT-CRC Error Detection**
  - **Interrupt on End of Message**
  - **IBM Retail Store Loop Mode**
- **Byte Control Protocol: IBM BISYNC**
  - **Special Character Generation: DLE, SYNC**
  - **Special Character Detection: DLE, SYNC, SOH, STX, ITB, ETB, ETX, ENQ**
  - **ASCII or EBCDIC**
  - **Normal and Transparent Text Mode**
  - **8-Bit Character Length**
  - **Automatic DLE Stuff-Stripping in Transparent Mode**
  - **Automatic Fill Character Insertion with Selectable Stripping**
  - **Selectable Leading Pad Transmission (Hex 55, 55)**
  - **CCITT, CRC-16 or VRC/LRC Error Detection**
  - **Interrupt on End of Message**
- **Byte Control Protocols: DDCMP and Other Custom BCP**
  - **Programmable SYNC Character**
  - **5- to 8-Bit Character Length**
  - **Selectable Leading Pad Transmission (Hex 55, 55)**
  - **Selectable CRC Error Detection**
  - **Automatic Fill Character Insertion with Selectable Stripping**
  - **CCITT, CRC-16, CRC-12, or (Odd/Even) VRC/LRC**
- **Directly Addressable Parameter Control Registers: Mode, SYNC/Address, Transmitter Control, and Receiver Control**
  - **Separate Addressable Status and Data Registers for Receiver and Transmitter**
  - **Modem Handshake Signals: $\overline{RTS}$, $\overline{CTS}$, $\overline{DTR}$, $\overline{DSR}$, and $\overline{CD}$**
  - **NRZ or NRZI (Complement on Zero) Serial Data**
  - **Full or Half-Duplex Operation**
  - **Self-Test Loop-Back Mode**
  - **8- or 16-Bit Bidirectional 3 State Data Bus**
  - **40 Pin Ceramic or Plastic Package**

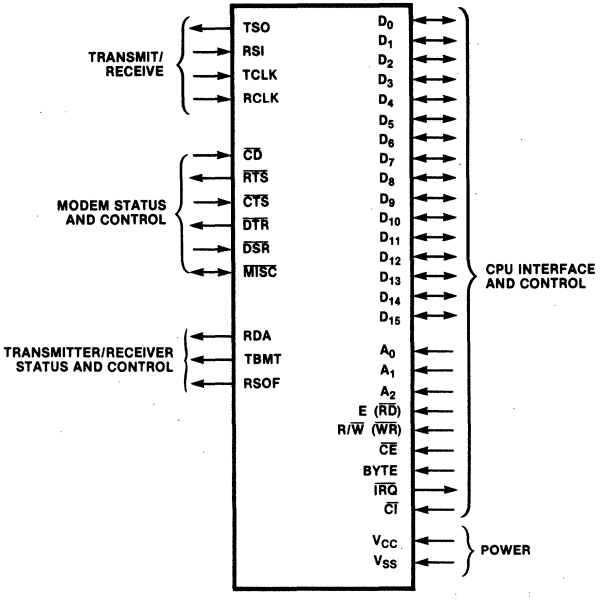
5

Connection Diagram

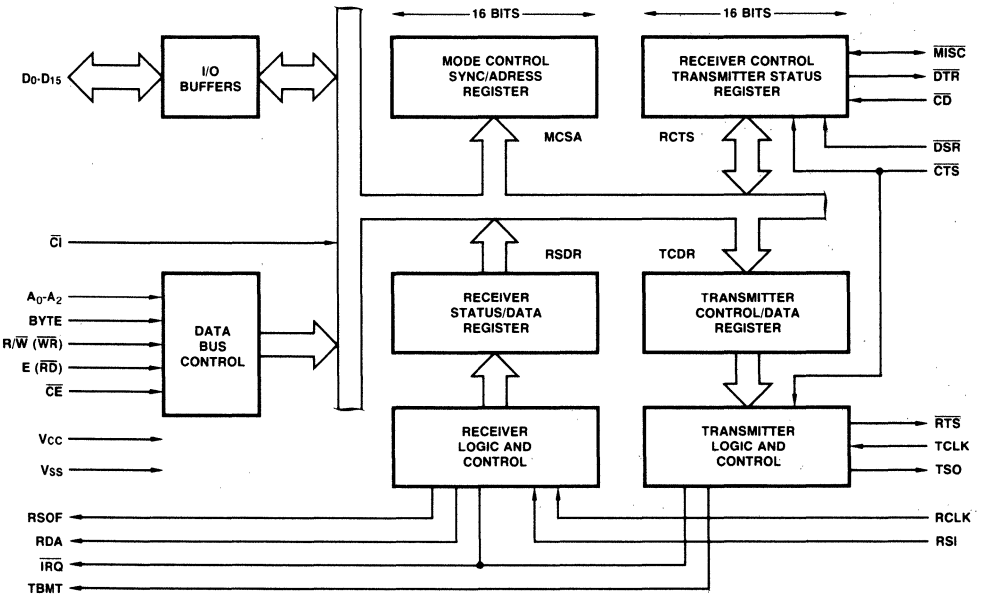


*F68456 Designation
†F38456 Designation

Signal Functions



Block Diagram



The F38456/F68456 signal functions are described in the following table.

Mnemonic	Pin No.	Name	Description
Transmit/Receive			
TSO	1	Transmitter Serial Output	This output signal is the transmitted serial data. Data changes on the positive going edge of TCLK.
RSI	38	Received Serial Input	This input signal is the received serial data. Data changes on the negative going edge of RCLK.
TCLK	2	Transmitter Clock	Timing of the transmitter logic is provided by this input signal. Frequency is the same as the transmitted baud rate.
RCLK	39	Receiver Clock	Timing for the receiver logic is provided by this input. Frequency is the same as the received baud rate.
Modem Status/Control			
$\overline{CD}$	27	Carrier Detect	This input is general-purpose in nature. It can be tested by reading the transmitter status register.
$\overline{RTS}$	3	Request to Send	This output is used with clear to send to enable the transmitter. It may be set low by programming the appropriate bit of the transmitter control register.
$\overline{CTS}$	26	Clear to Send	This input signal is used with request to send to enable the transmitter. It can be tested by reading the transmitter status register.
$\overline{DTR}$	23	Data Terminal Ready	This is a general-purpose output. It can be set low by programming the appropriate bit of the receiver control register.
$\overline{DSR}$	36	Data Set Ready	This is a general-purpose input. It can be tested by the CPU by reading the transmitter status register.
$\overline{MISC}$	22	Miscellaneous	This is a general-purpose input/output. It can be set low by programming the appropriate bit of the register; it can be tested by the CPU by reading the receiver control register.
Transmitter/ Receiver Status/Control			
RDA	14	Receiver Data Available	A high level on this output signal indicates an assembled character is in the receiver buffer. RDA is reset on the trailing edge of enable when the receiver buffer is read by the CPU.
TBMT	25	Transmitter Buffer Empty	A high level indicates the device is ready to receive new data and/or control information from the CPU. This output signal is reset on the leading edge of the first transmitter clock; it follows the trailing edge of enable when the transmitter buffer is loaded.

Mnemonic	Pin No.	Name	Description
Transmitter/ Receiver Status/Control			
RSOF	37	Received Sync or Flag	RSOF is high for one receiver clock period when a received sync or flag character is detected on this output signal.
CPU Interface and Control			
D_0 - D_{15}	6 - 13 28 - 35	Data Bus	These are 16 bidirectional input/output data lines which control information to and from the CPU. D_0 - D_7 can be wired to D_8 - D_{15} for use as an 8-bit data bus.
A_0 - A_2	15 - 17	Register Address	These input signals select internal data, status, and control registers. They may be selected as 8- or 16-bit registers.
E	19	Enable (F6456)	A strobe on this input causes information transfer between the data bus and the addressed register when the chip enable input is low.
$\overline{RD}$	19	Read Pulse (F38456)	A negative pulse on this input with address causes chip enable to transfer the data bus information to the addressed register.
R/$\overline{W}$	5	Read/Write (F68456)	A high level on this input allows data from the addressed register to be output to the data bus. A low level allows data from the bus to be loaded into the addressed register.
$\overline{WR}$	5	Write Pulse (F38456)	A negative pulse on this input with address causes chip enable to transfer the data bus information to the addressed register.
$\overline{CE}$	21	Chip Enable	A low level on this input signal enables a data bus transfer with enable.
BYTE	18	Byte	A high level on this input signal indicates an 8-bit data bus. A low level indicates a 16-bit bus.
$\overline{IRQ}$	24	Interrupt Request	This output goes low to indicate a change in the internal status of the device. The status bits linked to this output are receiver overrun (ROVR), received end-of-message (REOM) and transmitter underrun (TUR). IRQ is reset on the trailing edge of enable when the associated status register is read.
$\overline{CI}$	4	Chip Initialize	A low level initializes the internal control registers and timing on this input signal.
Power			
V_{DD}	40	Power Supply	Power supply input: +5 V ±5%
V_{SS}	20	Ground	Ground: 0 V reference

FAIRCHILD
A Schlumberger Company

F68488
General-Purpose Interface Adapter

Microprocessor Product

Description

The F68488 General-Purpose Interface Adapter (GPIA) provides the means to interface between an IEEE-488 standard instrument bus and the F6800. The 488 instrument bus provides a means for controlling and moving data from complex systems of multiple instruments.

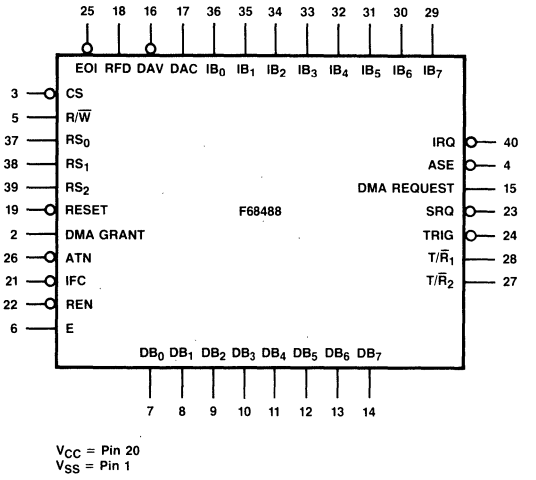
The F68488 will automatically handle all handshake protocol needed on the instrument bus.

- **Single or Dual Primary Address Recognition**
- **Secondary Address Capability**
- **Complete Source and Acceptor Handshakes**
- **Programmable Interrupts**
- **RFD Hold-Off to Prevent Data Overrun**
- **Operates with DMA Controller**
- **Serial and Parallel Polling Capability**
- **Talk-Only or Listen-Only Capability**
- **Selectable Automatic Features to Minimize Software**
- **Synchronization Trigger Output**
- **F6800 Bus Compatible**

Pin Names

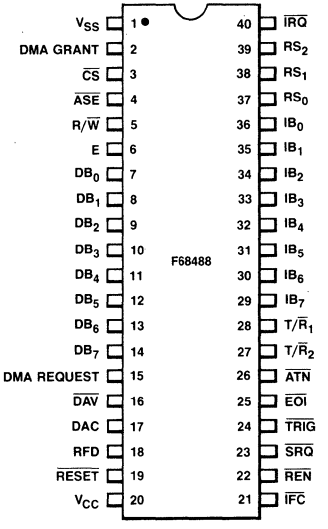
DB_0-DB_7	Bidirectional Data Lines
$\overline{CS}$	Chip Select Input
$R/\overline{W}$	Read/Write Input
RS_0, RS_1, RS_2	Register Select Inputs
$\overline{IRQ}$	Interrupt Request Output
$\overline{RESET}$	Chip Reset Input
DMA GRANT	DMA Transfer in Progress Input
DMA REQUEST	DMA Transfer Ready Output
$\overline{ASE}$	Address Switch Enable Output
IB_0-IB_7	Bidirectional ASCII Bus
DAC	Bidirectional Data Accepted Line
RFD	Bidirectional Ready for Data Line
$\overline{DAV}$	Bidirectional Data Valid Line
$\overline{ATN}$	Attention Input
$\overline{IFC}$	Interface Clear Input
$\overline{SRQ}$	Service Request Output
$\overline{REN}$	Remote Enable Input
$\overline{EOI}$	Bidirectional End or Identify Line
$\overline{TRIG}$	Group Execute Trigger Output
$T/\overline{R}_1$, $T/\overline{R}_2$	Transmit/Receive Control Outputs
E	Enable Clock Input
V_{SS}	Ground
V_{CC}	+5 V Power Supply

Logic Symbol



V_{CC} = Pin 20
V_{SS} = Pin 1

Connection Diagram
40-Pin DIP



(Top View)

5

Functional Description

The IEEE-488 instrument bus standard is a bit-parallel, byte-serial bus structure designed for communication to and from intelligent instruments. Using this standard, many instruments may be interconnected and remotely and automatically controlled or programmed. Data may be taken from, sent to or transferred between instruments. A bus controller dictates the role of each device by making the attention ($\overline{ATN}$) line true and sending talk or listen addresses on the instrument bus data lines; those devices that have matching addresses are activated. Device addresses are set into each GPIA from switches or jumpers on a pc board by a microprocessor as a part of the initialization sequence.

When the controller makes the $\overline{ATN}$ line true, instrument bus commands may also be sent to single or multiple GPIAs.

Information is transmitted on the instrument bus data lines under sequential control of the three handshake lines. No step in the sequence can be initiated until the previous step is completed. Information transfer can proceed as fast as the devices can respond, but no faster than the slowest device presently addressed as active. This permits several devices of different speeds to receive the same data concurrently.

The GPIA is designed to work with standard 488-bus driver ICs to meet the complete electrical specifications of the IEEE-488 bus. Additionally, a powered-off instrument may be powered-on without disturbing the 488 bus. With some additional logic, the GPIA could be used with other microprocessors.

The F68488 GPIA has been designed to interface the F6800 microprocessor with the complex protocol of the IEEE-488 instrument bus. Many instrument bus protocol functions are handled automatically by the GPIA and require no additional MPU action. Other functions require minimum MPU response due to a large number of internal registers conveying information on the state of the GPIA and the instrument bus.

Fig. 1. Functional Diagram

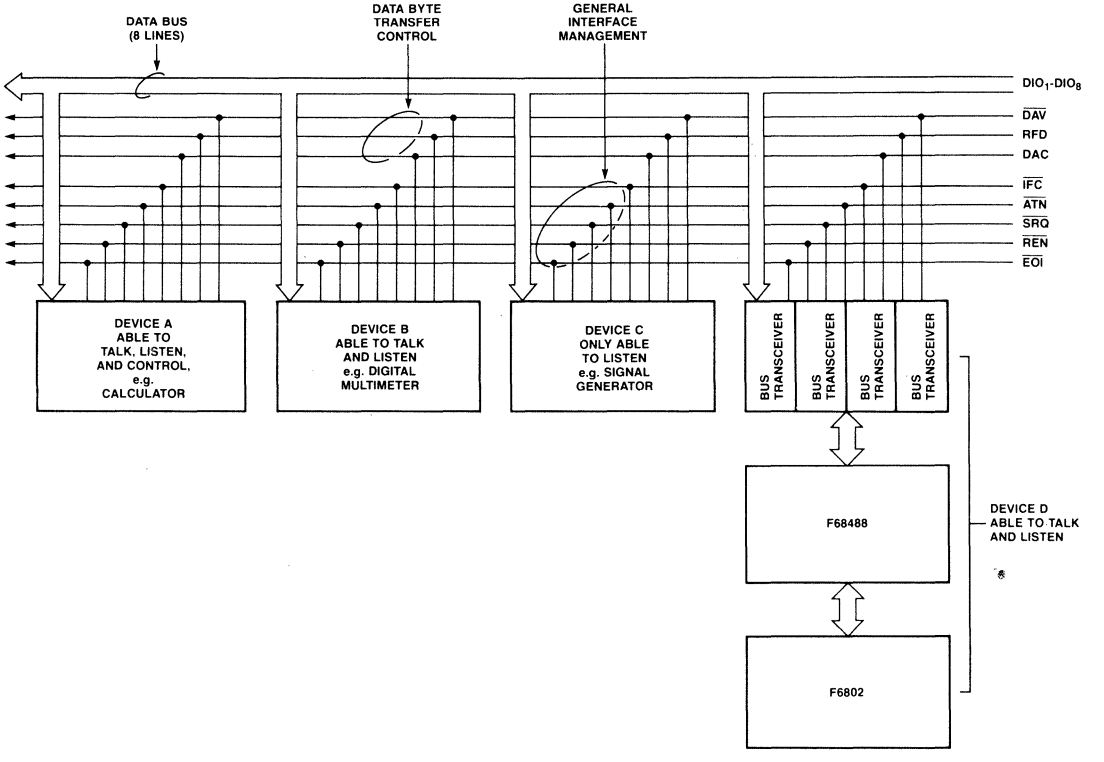
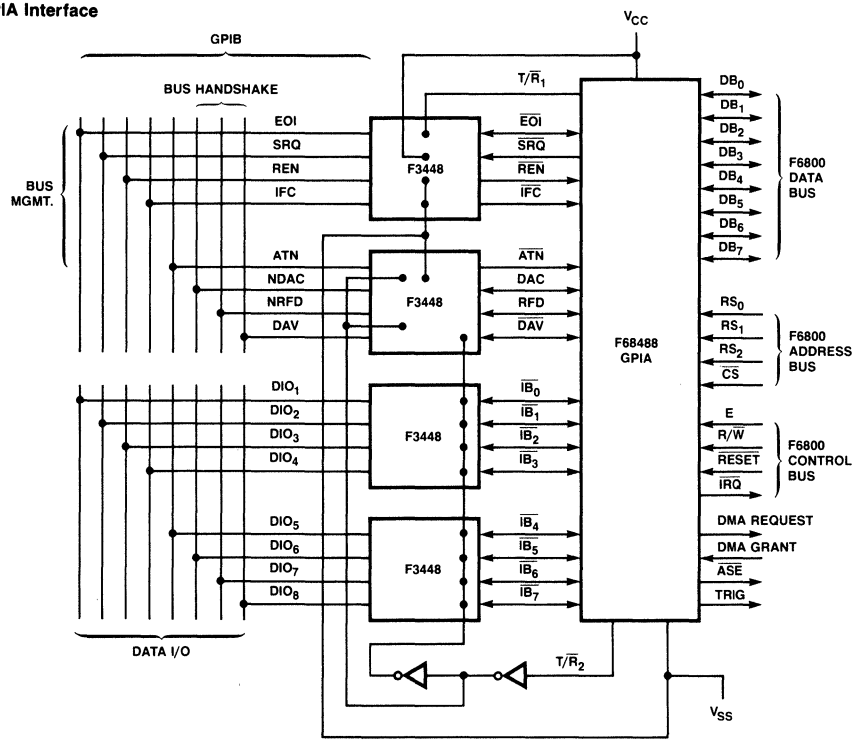


Fig. 2. F68488 GPIA Interface



GPIA/MPU Interface Signals
The F68488 interfaces to the F6800 MPU with an 8-bit bidirectional Data Bus, a Chip Select, Read/Write line, $\overline{\text{RESET}}$ line, three Register Select lines, an Interrupt Request line, two DMA Control lines, and an Address Switch Enable line.

GPIA Bidirectional Data (DB_0-DB_7) – The bidirectional data lines allow the transfer of data between the MPU and the GPIA. The data bus output drivers are 3-state devices that remain in the high-impedance (OFF) state except when the MPU performs a GPIA read operation. The Read/Write line is in the read state when the GPIA is selected for a read operation.

GPIA Chip Select ($\overline{\text{CS}}$) – This input signal is used to select the GPIA. The $\overline{\text{CS}}$ signal must be LOW for selection of the device. Chip select decoding will have to be accomplished with logic external to the chip.

GPIA Read/Write Line (R/$\overline{\text{W}}$) – This signal is generated by the MPU to control register access and direction of data transfer on the data bus. A LOW state on the GPIA Read/Write line allows for the selection of one of seven write-only registers when used in conjunction with the Register Select lines, RS_0, RS_1, RS_2. A HIGH state on the GPIA Read/Write line enables the selection of one of eight read-only registers when used in conjunction with the Register Select lines.

GPIA Register Select (RS_0, RS_1, RS_2) – The three register select lines are used to select the various registers inside the GPIA. These three lines are used in conjunction with the Read/Write line to select a particular register that is to be written to or read from. *Table 1* shows the register select coding.

Interrupt Request ($\overline{\text{IRQ}}$) – The $\overline{\text{IRQ}}$ output goes to the common interrupt bus for the MPU. This is an open drain output which is wire-ORed to the $\overline{\text{IRQ}}$ bus. The $\overline{\text{IRQ}}$ is set false (LOW) when an enabled interrupt occurs and stays false until the MPU reads from the interrupt status register.

$\overline{\text{RESET}}$ – The active-LOW $\overline{\text{RESET}}$ input is used to initialize the device during power-on start-up. The $\overline{\text{RESET}}$ line will be driven by an external power-on reset circuit.

DMA Control Lines (DMA Grant, DMA Request) – The DMA Request line is used to signal waiting data when Byte In (BI) Byte Out (BO) is set HIGH for a DMA controller. The DMA Request line is set HIGH if either the BI or BO interrupt flag is set in the interrupt status register (R0W) and the corresponding bits in the interrupt mask register (R0R) are set true. The DMA Request line is cleared when the DMA Grant is made true. The DMA Grant line is used to signal the GPIA that the DMA controller has control of the MPU data and address lines. The DMA Grant line must be grounded when not in use.

Trigger ($\overline{\text{TRIG}}$) – The TRIG pin provides an output corresponding to the GET and fget commands. A hardware or software reset places this output at a LOW level. The trigger output can be programmed HIGH by either of two methods:

1. Setting fget (bit 0 of R3W) by the MPU causes the trigger output to be set. It remains set until the fget bit is programmed LOW or until a reset occurs.
2. The trigger output is set upon reception of a GET command from the controller. It is reset when the GPIA moves out of the device trigger active state (DTAS); i.e., when GET, LADS, or ACDS occur.

Address Switch Enable ($\overline{\text{ASE}}$) – The $\overline{\text{ASE}}$ output is used to enable the device address switch 3-state buffers to allow the instrument address switches to be read on the MPU bus.

Enable Input (E) – The E input is normally a derivative of the MPU $\phi 2$ clock.

F68488-GPIA/488 Interface Bus Signals
The GPIA provides a set of 18 interface signal lines between the F6800 and the IEEE-488 Standard bus.

Signal Lines ($\overline{\text{IB}}_0$-$\overline{\text{IB}}_7$) – These bidirectional lines allow for the flow of 7-bit ASCII interface messages and device-dependent messages. Data appears on these lines in a bit-parallel, byte-serial form. These lines are buffered by the transceivers and applied to the 488 bus (DIO_1-DIO_8).

Byte Transfer Lines (DAC, RFD, $\overline{\text{DAV}}$) – These lines allow for proper transfer of each data byte on the bus between sources and acceptors. The RFD line goes passively true to indicate that all acceptors are ready for data. A source will indicate the data is valid by pulling $\overline{\text{DAV}}$ LOW. Upon the reception of valid data by all acceptors, DAC will go passively true to indicate that the data has been accepted by all acceptors.

Bus Management Lines ($\overline{\text{ATN}}$, $\overline{\text{IFC}}$, $\overline{\text{SRQ}}$, $\overline{\text{REN}}$, $\overline{\text{EOI}}$) – These lines are used to manage an orderly flow of information across the interface lines.

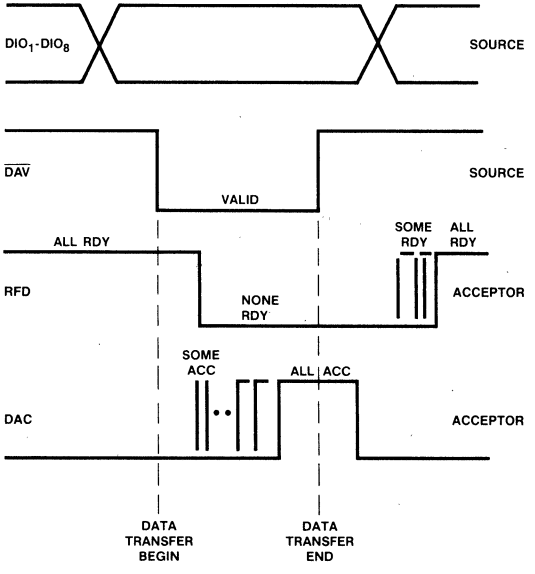
Attention ($\overline{\text{ATN}}$) – The $\overline{\text{ATN}}$ signal is sent true over the interface to disable current talkers and listeners, freeing the

Table 1 Register Access

RS_2	RS_1	RS_0	R/W	Register Title	Register Symbol
0	0	0	1	Interrupt Status	R0R
0	0	0	0	Interrupt Mask	R0W
0	0	1	1	Command Status	R1R
0	0	1	0	Unused	
0	1	0	1	Address Status	R2R
0	1	0	0	Address Mode	R2W
0	1	1	1	Auxiliary Command	R3R
0	1	1	0	Auxiliary Command	R3W
1	0	0	1	Address Switch*	R4R
1	0	0	0	Address	R4W
1	0	1	1	Serial Poll	R5R
1	0	1	0	Serial Poll	R5W
1	1	0	1	Command Pass-Through	R6R
1	1	0	0	Parallel Poll	R6W
1	1	1	1	Data-In	R7R
1	1	1	0	Data-Out	R7W

*External to F68488

Fig. 3. Source and Acceptor Handshake



signal lines ($\overline{IB}_0$-$\overline{IB}_7$). During the $\overline{ATN}$ active state, devices monitor the DIO_1-DIO_8 lines for addressing or an interface command. Data flows on the DIO_1-DIO_8 when $\overline{ATN}$ is inactive (HIGH).

Interface Clear ($\overline{IFC}$) – The $\overline{IFC}$ signal is used to put the interface system into a known quiescent state.

Service Request ($\overline{SRQ}$) – The $\overline{SRQ}$ signal is used to indicate a need for attention in addition to requesting an interruption in the current sequence of events. This indicates to the controller that a device on the bus is in need of service.

Remote Enable ($\overline{REN}$) – The $\overline{REN}$ signal is used to select one of two alternate sources of devices programming data, local, or remote control.

END or Identify ($\overline{EOI}$) – The $\overline{EOI}$ signal is used to signal the end of a multiple byte transfer sequence and, in conjunction with $\overline{ATN}$, executes a parallel polling sequence.

Transmit/Receive Control Signals ($T/\overline{R}_1$, $T/\overline{R}_2$) – These two signals are used to control the bus transceivers that drive the interface bus. It is assumed that transceivers equivalent to the F3447 or F3448 will be used, where each transceiver has a separate Transmit/Receive control pin. These pins can support one TTL load each. The outputs can then be grouped as shown in *Figure 1* with $\overline{SRQ}$ hardwired HIGH to transmit. The $\overline{REN}$, $\overline{IFC}$, and $\overline{ATN}$ lines are hardwired LOW to receive. The $\overline{EOI}$ line is controlled by $T/\overline{R}_1$ through the bus transceiver, allowing it to transmit or receive. The $T/\overline{R}_1$ line operates exactly as $T/\overline{R}_2$, except during the parallel polling sequence. During parallel poll, $\overline{EOI}$ will be made an input by $T/\overline{R}_1$ while the $\overline{DAV}$ and $\overline{IB}_0$-$\overline{IB}_7$ lines are outputs.

GPIA Internal Controls & Registers

There are 15 locations accessible to the MPU data bus that are used for transferring data to control the various functions of the device and provide current device status. Seven of these registers are write-only and eight are read-only. The various registers are accessed according to the three least significant bits of the MPU address bus and the status of the Read/Write line. One of the 15 registers is external to the device, but an address switch register is provided for reading the address switches. *Table 2* shows actual bit contents of each of the registers.

Data-In Register R7R – The data-in register is an actual 8-bit storage register used to move data from the interface bus when the device is a listener. Reading the register does not destroy information in the data-out register. The DAC (data accepted) line will remain LOW until the MPU removes the byte from the data-in register. The device will automatically finish the handshake by allowing DAC to go HIGH. In RFD (ready for data) hold-off mode, a new handshake is not initiated until a command is sent allowing the device to release hold-off. This will delay a talker until the available information has been processed.

Data-In Register (Read-Only)

DI_7	DI_6	DI_5	DI_4	DI_3	DI_2	DI_1	DI_0

DI_0-DI_7 – correspond to DIO_1-DIO_8 of the 488-1975 standard and $\overline{IB}_0$-$\overline{IB}_7$ of the F68488

Data-Out Register R7W – The data-out register is an actual 8-bit storage register used to move data out of the device onto the interface bus. Reading from the data-in register has no effect on the information in the data-out register. Writing to the data-out register has no effect on the information in the data-in register.

Data-Out Register (Write-Only)

DO_7	DO_6	DO_5	DO_4	DO_3	DO_2	DO_1	DO_0

DO_0-DO_7 – correspond to DIO_1-DIO_8 of the 488-1975 standard and $\overline{IB}_0$-$\overline{IB}_7$ of the F68488

Interrupt Mask Register R0W – The interrupt mask register is a 7-bit storage register used to select the particular events that will cause an interrupt to be sent to the MPU. The seven control bits may be set independently of each other. If dsel (bit 7 of the address mode register) is set HIGH, CMD (bit 2) will interrupt SPAS or RLC. If dsel is set LOW, CMD will interrupt on UACG, UUCG, and DCAS in addition to RLC and SPAS. The command status register R1R may then be used to determine which command caused the interrupt. Setting GET (bit 5) allows an interrupt to occur on the Group Execute Trigger Command. The END bit (bit 1) allows an interrupt to occur if $\overline{EOI}$ is true (LOW) and $\overline{ATN}$ is false (HIGH). The APT bit (bit 3) allows an interrupt to occur indicating that a secondary address is available to be examined by the MPU if apte (bit 0 of the address mode register) is enabled, listener or talker primary address is received, and a Secondary Command Group is received. A typical response for a valid secondary address would be to set msa (bit 3 of the auxiliary command register) and dacr (bit 4 of the auxiliary command register), releasing the DAC handshake. The BI bit (bit 0) indicates that a data byte is waiting in the data-in register. BI is set HIGH when the data-in register is full. The BO bit (bit 6) indicates that the data-out register is empty. BO is set when the data-out register is empty. The IRQ bit (bit 7) allows any interrupt to be passed to the MPU.

Interrupt Mask Register (Write-Only)

IRQ	BO	GET	X	APT	CMD	END	BI

IRQ – Mask bit for $\overline{IRQ}$ Output

Table 2 Internal Register Contents

Register Name	Mnemonic	Bit 7	6	5	4	3	2	1	0
Interrupt Mask Register	R0W	IRQ	BO	GET		APT	CMD	END	BI
Interrupt Status Register	R0R	INT	BO	GET		APT	CMD	END	BI
Command Status Register	R1R	UACG	REM	LOK		RLC	SPAS	DCAS	UUCG
Unused	R1W								
Address Status Register	R2R	ma	to	lo	ATN	TACS	LACS	LPAS	TPAS
Address Mode Register	R2W	dsel	to	lo		hlde	hlda		apte
Auxiliary Command Register	R3R	Chip	DAC	DAV	RFD	msa	rtl	ulpa	fget
	R3W	RESET	rfdr	feoi	dacr	msa	rtl	dacd	fget
Address Switch Register	R4R	UD_3	UD_2	UD_1	AD_5	AD_4	AD_3	AD_2	AD_1
Address Register	R4W	lsbe	dal	dat	AD_5	AD_4	AD_3	AD_2	AD_1
Serial Poll Register	R5R	S_8	SRQS	S_6	S_5	S_4	S_3	S_2	S_1
	R5W	S_8	rsv	S_6	S_5	S_4	S_3	S_2	S_1
Command Pass-through Register	R6R	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
Parallel Poll Register	R6W	PPR_8	PPR_7	PPR_6	PPR_5	PPR_4	PPR_3	PPR_2	PPR_1
Data-In Register	R7R	DI_7	DI_6	DI_5	DI_4	DI_3	DI_2	DI_1	DI_0
Data-Out Register	R7W	DO_7	DO_6	DO_5	DO_4	DO_3	DO_2	DO_1	DO_0

BO – Interrupt on Byte Output

GET – Interrupt on Group Execute Trigger

APT – Interrupt on Secondary Address Pass-Through

CMD – Interrupt on SPAS + RLC + $\overline{dsel}$ (DCAS + UUCG + UACG)

END – Interrupt on $\overline{EOI}$ and $\overline{ATN}$

BI – Interrupt on Byte Input

Interrupt Status Register R0R – The interrupt status register is a 7-bit storage register that corresponds to the interrupt mode register with an additional bit, INT (bit 7). Except for the INT bit, the other bits in the status register are set regardless of the state of the interrupt mode register when the corresponding event occurs. The IRQ (MPU Interrupt) is cleared when the MPU reads from the register. The INT bit is the logical OR of the other six bits ANDed with the respective bit of R0W.

Interrupt Status Register (Read-Only)

INT	BO	GET	X	APT	CMD	END	BI

INT – Logical OR of all other bits in this register ANDed with the respective bits in the interrupt mask register

BO – A byte of data has been output.

GET – A Group Execute Trigger has occurred.

APT – An Address Pass-Through has occurred.

CMD – SPAS + RLC + $\overline{dsel}$ (DCAS + UUCG + UACG) has occurred.

END – An $\overline{EOI}$ has occurred with $\overline{ATN}$ = HIGH.

BI – A byte has been input.

Serial Poll Register R5R/W – The serial poll register is an 8-bit storage register that can be both written into and read from by the MPU. It is used for establishing the status byte that the device sends out when it is serial poll enabled. Status may be placed in bits 0 through 5 and bit 7. Bit 6 rsv (request for service) is used to drive the logic that controls the $\overline{SRQ}$ line on the bus telling the controller that service is needed. This same logic generates the service request state (SRQS) signal that is substituted in the bit 6 position when the status byte is read by the MPU $\overline{IB_0}$-$\overline{IB_7}$. In order to initiate an rsv (request for service), the MPU sets bit 6 true (generating an rsv signal) and this in turn causes the device to pull down the $\overline{SRQ}$ line. The SRQS signal is the same as rsv when SPAS is false. Bit 6, as read by the MPU, will be the SRQS.

Serial Poll Register (Read)

S_8	SRQS	S_6	S_5	S_4	S_3	S_2	S_1

S_1-S_8 – Status bits

SRQS – Bus is in service request status state

Serial Poll Register (Write)

S_8	rsv	S_6	S_5	S_4	S_3	S_2	S_1

S_1-S_8 – Status bits

rsv – Generate a service request

Parallel Poll Register R6W – This register will be loaded by the MPU, and the complement of the bits in this register will be delivered to the instrument bus (IB_0-IB_7) during PPAS (Parallel Poll Active State). This register powers up in the PP0 (Parallel Poll No Capability) state. The reset bit (auxiliary command register bit 7) will clear this register to the PP0 state.

The parallel poll interface function is executed by this device using the PP2 subset (Omit Controller Configuration Capability). The controller cannot directly configure the parallel poll output of this device. This must be done by the MPU. The controller will be able to configure the parallel poll indirectly by issuing an addressed command that has been defined in the MPU software.

Parallel Poll Register (Write-Only)

PP_8	PP_7	PP_6	PP_5	PP_4	PP_3	PP_2	PP_1

Bits delivered to bus during Parallel Poll Active State (PPAS)

Register powers-up in the PP0 state.

Parallel Poll is executed using the PP2 subset.

Address Mode Register R2W – The address mode register is a storage register with six bits for control: to, lo, hlde, hlda, dsel, and apte. The to bit (bit 6) selects the talker/listener and addresses the device to talk only. The lo bit (bit 5) selects the talker/listener and sets the device to listen only. The apte bit (bit 0) is used to enable the extended addressing mode. If apte is set LOW, the device goes from the TPAS (Talker Primary Address State) directly to the TADS (Talker Addressed State). If apte is set HIGH and the secondary address is valid, set msa true. The hlda bit (bit 2) holds off RFD (Ready for Data) on all data until rfdr is set true. The hlde bit (bit 3) holds off RFD on $\overline{EOI}$ enabled (LOW) and $\overline{ATN}$ not enabled (HIGH). This allows the last byte in a block of data to be continually read as needed. Writing rfdr true (HIGH) will release the handshake.

Address Mode Register (Write-Only)

dsel	to	lo	X	hdle	hdla	X	apte

dsel – Configure for automatic completion of handshake sequence on occurrence of GET, UACG, UUCG, SDC, or DCL commands

to – Set to talk-only mode

lo – Set to listen-only mode

hdle – Hold-off RFD on end

hdla – Hold-off RFD on all data

apte – Enable the address pass-through feature

Address Status Register R2R – The address status register is not a storage register, but is simply an 8-bit port used to couple internal signal modes to the MPU bus. The status flags represented here are stored internally in the logic of the device. These status bits indicate the addressed state of the talker/listener as well as flags that specify whether the device is in the talk-only or listen-only mode. The ma signal is true when the device is in:

TACS – Talker Active State

TADS – Talker Addressed State

LACS – Listener Active State

LADS – Listener Addressed State

SPAS – Serial Poll Active State

ATN – Bit 4 contains the condition of the attention line

Address Status Register (Read-Only)

ma	to	lo	ATN	TACS	LACS	LPAS	TPAS

ma – My address has occurred.

to – The talk-only mode is enabled.

lo – The listen-only mode is enabled.

ATN – The Attention command is asserted.

TACS – GPIA is in the Talker Active State.

LACS – GPIA is in the Listener Active State.

LPAS – GPIA is in the Listener Primary Addressed State.

TPAS – GPIA is in the Talker Primary Addressed State.

Address Switch Register R4R – The address switch register is external to the device. There is an enable line ($\overline{ASE}$) to be used to enable 3-state drivers connected between the address switches and the MPU. When the MPU addresses the address switch register, the enable line directs the switch information to

be sent to the MPU. The five least significant bits of this 8-bit register are used to specify the bus address of the device, and the remaining three bits may be used at the discretion of the user. The most probable use of one or two of the bits is for controlling the listen-only or talk-only functions.

Address Switch Register (Read-Only)

UD_3	UD_2	UD_1	AD_5	AD_4	AD_3	AD_2	AD_1

AD_1-AD_5 – Device Address

UD_1-UD_3 – User Definable Bits

When this register is addressed, the $\overline{ASE}$ pin is set to allow external address switch information to be read from a bus device.

Address Register R4W – The address register is an 8-bit storage register. The purpose of this register is to carry the primary address of the device. The primary address is placed in the five least significant bits of the register. If external switches are used for device addressing, these are normally read from the address switch register and then placed in the address register by the MPU.

The AD_1-AD_5 bits (0 - 5) are for the device address. The lsbe bit (bit 7) is set to enable the dual primary addressing mode. During this mode, the device will respond to two consecutive addresses; one address with AD_1 equal to 0 and the other address with AD_1 equal to 1. For example, if the device address is $0F, the dual primary addressing mode would allow the device to be addressed at both $0F and $0E. The dal bit (bit 6) is set to disable the listener and the dat bit (bit 5) is set to disable the talker.

This register is cleared by the $\overline{RESET}$ input only (not by the reset bit of the auxiliary command register, bit 7). When $\overline{ATN}$ is enabled and the primary address is received on the $\overline{IB}_0$-$\overline{IB}_7$ lines, the F68488 will set bit 7 of the address status register (MA). This places the F68488 in the TPAS or LPAS.

When $\overline{ATN}$ is disabled, the GPIA may go to one of three states: TACS, LACS or SPAS.

Address Register (Write-Only)

lsbe	dal	dat	AD_5	AD_4	AD_3	AD_2	AD_1

lsbe – Enable dual primary addressing mode

dal – Disable the listener

dat – Disable the talker

AD_1-AD_5 – Primary device address, usually read from address switch register

Register is cleared by $\overline{RESET}$ input pin only.

Auxiliary Command Register R3R/W – Bit 7, reset, initializes the device to the following states (reset is set true by external $\overline{RESET}$ input pin and by writing into the register from the MPU):

SIDS – Source Idle State

AIDS – Acceptor Idle State

TIDS – Talker Idle State

LIDS – Listener Idle State

LACS – Listener Active State

PPIS – Parallel Poll Idle State

PUCS – Parallel Poll Unaddressed to Configure State

PP0 – Parallel Poll No Capability

The rfdr (release RFD handshake) bit (bit 6) allows for completion of the handshake that was stopped by RFD (Ready For Data) hold-off commands hlda and hlde.

The fget (force group execute trigger) bit (bit 0) has the same effect as the GET (Group Execute Trigger) command from the controller. (IEEE STD 488 p. 39.)

The rtl (return to local) bit (bit 2) allows the device to respond to local controls and the associated device functions are operative.

The dacr (release DAC handshake) bit (bit 4) is set HIGH to allow DAC to go passively true. This bit is set to indicate that the MPU has examined a secondary address or an undefined command.

The ulpa (upper/lower primary address) bit (bit 1) will indicate the state of bit 0 on the DIO_1-DIO_8 bus lines at the time the last primary address was received. This bit can be read but not written by the MPU.

The msa (valid secondary address) bit (bit 3) is set true (HIGH) when TPAS (Talker Primary Addressed State) or LPAS (Listener Primary Addressed State) is true. The device will become addressed to listen or talk.

The primary address must have been previously received.

The RFD, DAV, and DAC (Ready for Data, Data Valid, and Data Accepted) bits assume the same state as the corresponding signal on the F68488 package pins. The MPU may only read these bits. These signals are not synchronized with the MPU clock.

The dacd (data accept disable) bit (bit 1) set HIGH by the MPU will prevent automatic handshake on addresses or commands. The dacr bit is used to release the handshake.

The feoi (forced end or identify) bit (bit 5) tells the device to send $\overline{EOI}$ LOW with the next data byte transmitted. The $\overline{EOI}$ line is then returned HIGH after the next byte is transmitted. NOTE: The following signals are not stored but revert to a false (LOW) level one clock cycle (MPU $\phi 2$) after they are set true (HIGH):

1. rfdr
2. feoi
3. dacr

These signals can be written but not read by the MPU.

Auxiliary Command Register

reset	rfdr	feoi	dacr	msa	rtl	dacd	fget	W
reset	DAC	DAV	RFD	msa	rtl	ulpa	fget	R

reset – Initialize the chip to the following status:

1. All interrupts cleared
2. Following bus states are in effect: SIDS, AIDS, TIDS, LIDS, LOCS, PPIS, PUCS, and PP0
3. Bit is set by $\overline{RESET}$ input pin.

DAC – Corresponds to Data Accepted signal on F68488 package pins

DAV – Corresponds to Data Valid signal on F68488 package pins

RFD – Corresponds to Ready For Data signal on F68488 package pins

msa – If GPIA is in LPAS or TDAS, setting msa will force GPIA to LADS or TADS.

rtl – Return to local if local lockout is disabled

ulpa – State of LSB of the address received on the DIO_{1-8} bus lines

fget – Force Group Execute Trigger Command from controller has occurred.

rfdr – Complete handshake stopped by RFD hold-off

feoi – Set $\overline{EOI}$ true, clears after next byte transmitted

dacr – MPU has examined an undefined command or secondary address.

dacd – Prevents automatic handshake on addresses or commands

Command Status Register R1R – The command status register flags commands or states as they occur. These flags or states are simply coupled onto the MPU bus from internal storage nodes.

These are five major address commands. REM shows the remote/local state of the talker/listener.

The RLC bit (bit 3) is set whenever a change of state of the remote/local flip-flop occurs and reset when the command status register is read.

The DCAS bit (bit 1) indicates that either the device clear or selected device clear has been received, activating the device clear function.

The SPAS bit (bit 2) indicates that the SPE command has been received, activating the device serial poll function.

The UACG bit (bit 7) indicates that an undefined address command has been received and, depending on programming, the MPU decides whether to execute or ignore it.

The UUCG bit (bit 0) indicates that an undefined universal command has been received.

Command Status Register (Read)

UACG	REM	LOK	X	RLC	SPAS	DCAS	UUCG

UACG – Undefined Address Command

REM – Remote Enabled

LOK – Local Lockout Enabled

RLC – Remote Local State Changed

SPAS – Serial Poll Active State is in effect.

DCAS – Device Clear Active State is in effect.

UUCG – Undefined Universal Command

Command Pass-Through Register R6R – The command pass-through register is an 8-bit port with no storage. When this port is addressed by MPU, it connects the instrument data bus ($\overline{IB_0}$-$\overline{IB_7}$) to the MPU data bus DB_0-DB_7. This port can be used to pass commands and secondary addresses, that are not automatically interpreted, through to the MPU for inspection.

Command Pass-Through Register (Read Only)

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0

An 8-bit port used to pass commands and secondary addresses to the MPU that are not automatically interpreted by the GPIA.

Absolute Maximum Ratings

Voltage of any pin relative to ground	−0.3 V, +7.0 V
Operating Temperature (Ambient)	0°C, +70°C
Storage Temperature (Ambient)	−55°C, +150°C
Power Dissipation	1 W

Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = 0 to +70°C, unless otherwise noted

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
V_{IH}	Input HIGH Voltage	2.0			V	
V_{IL}	Input LOW Voltage			0.8	V	
I_{IN}	Input Leakage Current		1.0	2.5	μA	V_{IN} = 0 to 5.25 V
I_{TSI}	3-State (OFF State) Input Current D_0-D_7		2.0	10	μA	V_{IN} = 0.4 to 2.4 V
V_{OH}	Output HIGH Voltage D_0-D_7 Other Outputs	 2.4 2.4			V	 I_{Load} = −205 μA I_{Load} = −200 μA
V_{OL}	Output LOW Voltage D_0-D_7 IRQ			 0.4 0.4	V	 I_{Load} = 1.6 mA I_{Load} = 3.2 mA
I_{LOH}	Output Leakage Current (OFF State) $\overline{IRQ}$		 1.0	 10	μA	 V_{OH} = 2.4 V
P_D	Power Dissipation		600		mW	
C_{IN}	Input Capacitance D_0-D_7 All Other Inputs			 12.5 7.5	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz
C_{OUT}	Output Capacitance IRQ			5.0	pF	V_{IN} = 0, T_A = 25°C, f = 1.0 MHz

Bus Timing Characteristics
Read *(Figure 4)*

Symbol	Characteristic	Min	Typ	Max	Unit
$t_{cyc}E$	Enable Cycle Time	1.0		25	μs
PW_{EH}	Enable Pulse Width, HIGH	0.45			μs
PW_{EL}	Enable Pulse Width, LOW	0.43			μs
t_{AS}	Set-up Time, Address and R/W Valid to Enable Positive Transistion	160			ns
t_{DDR}	Data Delay Time			320	ns
t_H	Data Hold Time	10			ns
t_{AH}	Address Hold Time	10			ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input			25	ns
Write *(Figure 5)*					
$t_{cyc}E$	Enable Cycle Time	1.0			μs
PW_{EH}	Enable Pulse Width, HIGH	0.45		25	μs
PW_{EL}	Enable Pulse Width, LOW	0.43			μs
t_{AS}	Set-up Time, Address and R/W Valid to Enable Positive Transistion	160			ns
t_{DSW}	Data Set-up Time	195			ns
t_H	Data Hold Time	10			ns
t_{AH}	Address Hold Time	10			ns
t_{Er}, t_{Ef}	Rise and Fall Time for Enable Input			25	ns

5

Fig. 4 Bus Read Timing Characteristics
(Read Information from GPIA)

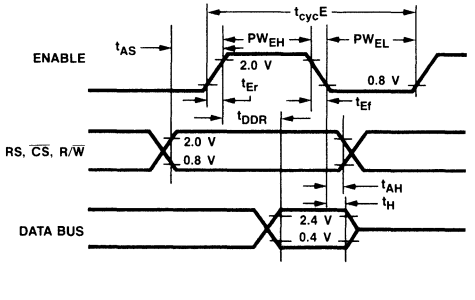
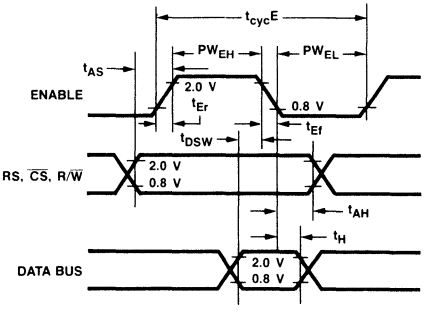


Fig. 5 Bus Write Timing Characteristics
(Write Information into GPIA)



Ordering Information

Speed	Order Code	Temperature Range
1.0 MHz	F68488P,S	0°C to +70°C
	F68488CP,CS	−40°C to +85°C
	F68488DL	−55°C to +85°C
	F68488DM	−55°C to +125°C
1.5 MHz	F68A488P,S	0°C to +70°C
	F68488CP,CS	−40°C to +85°C
2.0 MHz	F68B488P,S	0°C to +70°C

P = Plastic package, S = CER-DIP package

FAIRCHILD
A Schlumberger Company

Section 6
I³L Microprocessor Family

General

Fairchild has utilized bipolar Isoplanar Integrated Injection Logic (I^3L) VLSI proven technology to develop very fast 16-bit microprocessors. The F9445 is available now. Typical execution times for the F944520DM (20 MHz clock frequency over a $-55°C$ to $+125°C$ operating temperature range) are: Add in 0.3 μs and a full 16 × 16 bit in 3.5 μs. The high speed multiply and divide times of the F9445 make this device particularly well suited for real time control and signal processing applications.

Since the F9445 has been implemented in bipolar injection logic, it maintains full high speed performance at high temperature. Thus, it is an excellent processor for use in harsh environments. The F9445 is available in either dual-in-line (DIP) or JEDEC chip carrier packages. The processor is available fully screened per MIL-STD-883 Method 5004.

The F9445, which is supported by a family of peripheral chips, can address up to 64K words of memory, directly address 62 I/O devices, handle 16 levels of priority interrupt, and perform fast direct memory access. It supplies the signals necessary for operation in multiprocessor environment, and supports minicomputer-like console functions, including internal self-testing. The support devices, like the processor, are implemented in bipolar I^3L technology; their operating temperature range is also $-55°C$ to $+125°C$.

Fairchild provides a full range of design support for the F9445 16-bit processor. The Fairchild System-I (FS-I) Microprocessor Development Station provides a means for developing F9445 software. This system is fully supported by the IMDOS operating system. High level language compilers are currently available in FORTRAN and PASCAL, and in the future DOD Standard JOVIAL J73. The EMUTRAC option to the FS-I provides full in-circuit emulation and tracing of the F9445 system; it also provides simultaneous and interactive hardware and software development and debugging.

A complete F9445 microcomputer is available in the PEP-45. This powerful single board microcomputer is an excellent approach to becoming familiar with the F9445 family. The PEP-45 is useful as development tool, prototyping device, or as a standalone microcomputer. The PEP-45 has two serial I/O ports, onboard EPROM programmers, and meets IEEE bus standards.

Fairchild is currently developing the F9450 16-bit microprocessor. This bipolar I^3L VLSI device implements the full MIL-STD-1750A Instruction Set Architecture in a single monolithic microprocessor.

Also produced using I^3L technology are the F9414 Data Encryption Set and the F9423 FIFO Buffer Memory.

Instruction Set

Each 16-bit F9445 instruction word is divided into smaller sections, or fields, that specify the operation code and related actions, the CPU register conditions and registers, and the I/O device codes, and that derive memory location effective addresses.

The instruction set consists of the following types of instructions:

1. Arithmetic and logic instructions
2. Memory reference instructions
3. Stack manipulation instructions
4. I/O instructions
5. Control instructions

The F9445 instruction set is shown on the following pages. The assembly-language format of each instruction is shown on the left, followed by the name of the instruction and a symbolic description of its action. The corresponding bit pattern for each instruction is shown on the right side of the page.

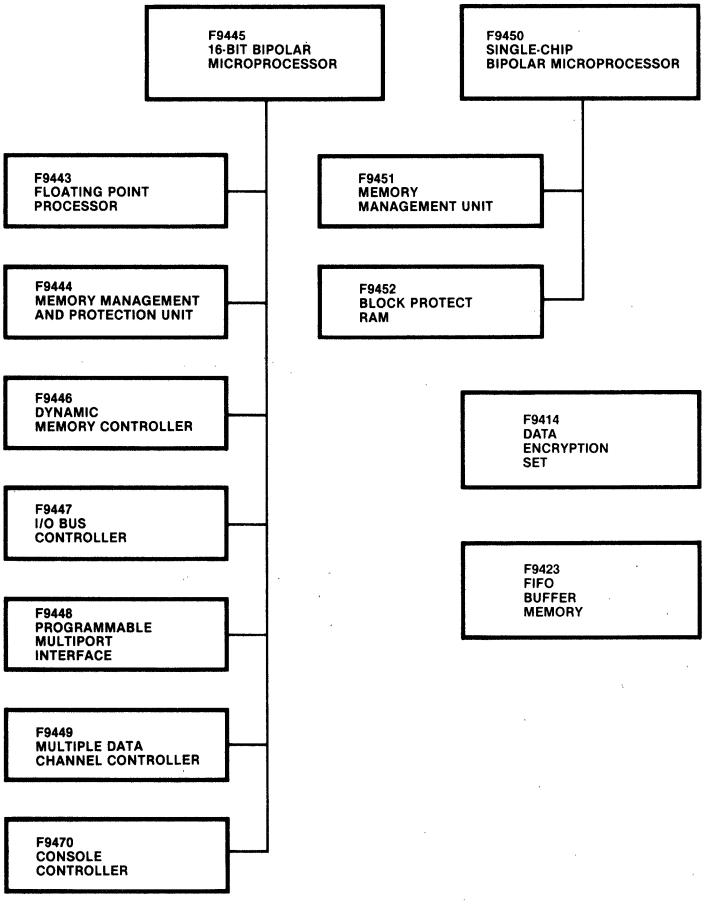
Assembly-language mnemonics and binary representations for instruction optional parts (within square brackets) and accumulator codes, to be inserted at the indicated places in the instructions, are shown following each group of instructions.

The required separator, indicated in the assembly-language formats by a square (□), may be entered as any number or combination of space or tab characters or a comma for the macro-assembler; the separator must be a single space for the PEPBUG-45 program.

Descriptions

Following is data that describes the members of the F9445 microprocessor family.

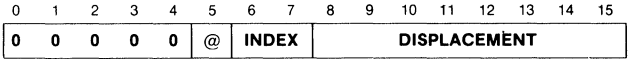
16-BIT I³L BIPOLAR MICROPROCESSOR FAMILY ORGANIZATION



Memory Reference Instructions

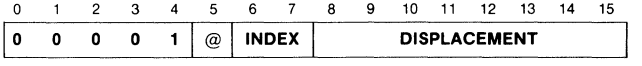
JMP□[@] displacement [,index]

Jump. Jump to effective address.



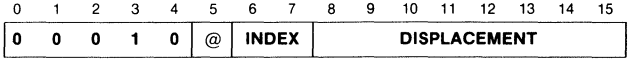
JSR□[@] displacement [,index]

Jump to Subroutine. Jump to subroutine at effective address: then return to PC saved in AC3.



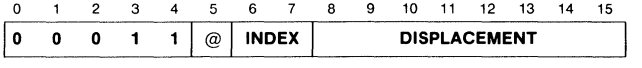
ISZ□[@] displacement [,index]

Increment and Skip if Zero. Increment (EA); if zero, skip next instruction.



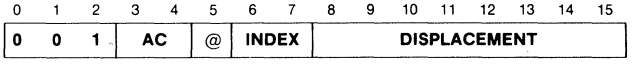
DSZ□[@] displacement [,index]

Decrement and Skip if Zero. Decrement (EA); if zero, skip next instruction.



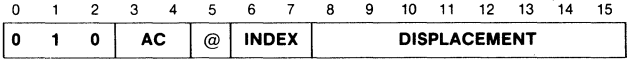
LDA□AC, [@] displacement [,index]

Load Accumulator. (EA) → AC.



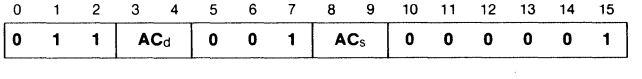
STA□AC, [@] displacement [,index]

Store Accumulator. AC → (EA).



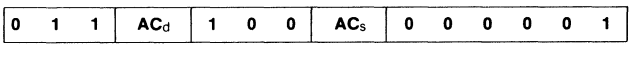
LDB□AC$_S$, AC$_d$,

Load Byte. (Byte Pointer) → AC$_{d\ 8-15}$, 0 → AC$_{d\ 0-7}$; LSB of byte pointer in AC$_S$ selects high-order byte if 0, low-order byte if 1.



STB□AC$_S$, AC$_d$

Store Byte. AC$_{d\ 8-15}$ → (Byte Pointer).



Effective Address Codes

Mnemonics		Bits			
@	Index	5	6	7	Effective Address
Omitted	0	0	0	0	EA = D; direct page zero
Omitted	1	0	0	1	EA = CA + D; relative to PC
Omitted	2	0	1	0	EA = AC2 + D; indexed by AC2
Omitted	3	0	1	1	EA = AC3 + D; indexed by AC3
@	0	1	0	0	EA = (D); indirect through page zero
@	1	1	0	1	EA = (CA + D); indirect relative to PC
@	2	1	1	0	EA = (AC2 + D); indirect relative to AC2
@	3	1	1	1	EA = (AC3 + D); indirect relative to AC3

Notes

D = Displacement; specified as absolute in current radix or relative via mnemonics.
CA = Current address or PC-1.
EA = Effective address.
(XX) = Contents of location XX, e.g. (EA) = Contents of effective address.
@ = Indirect address bit.
Byte Pointer (32K) = 16 bits of AC$_S$.
Byte Pointer (64K) = 17 bits of Carry and AC$_S$; upper 32K accessed when Carry = 1, lower 32K when Carry = 0.
□ = Required separator.

Accumulator Codes

	Bits		
Mnemonic	8	9	AC
0	0	0	AC0
1	0	1	AC1
2	1	0	AC2
3	1	1	AC3

Relative Displacement

Mnemonic	Meaning
.+D	Current location plus displacement
.—D	Current location minus displacement

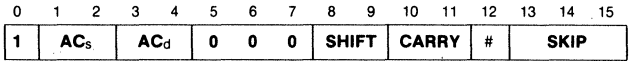
Note

D = Displacement in current radix.

Arithmetic and Logic Instructions

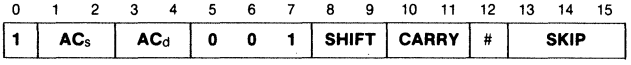
COM[carry][shift][#]□ACs, ACd [,skip]

Complement. $\overline{AC_S}$ → AC_d.



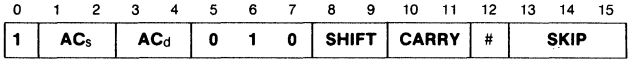
NEG[carry][shift][#]□ACs, ACd [,skip]

Negate. —AC_S → AC_d; affects Carry and Overflow flags.



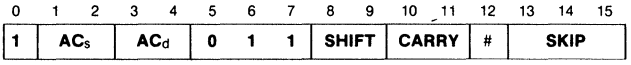
MOV[carry][shift][#]□ACs, ACd [,skip]

Move. AC_S → AC_d.



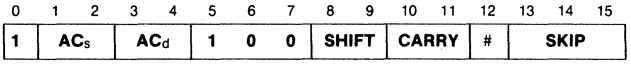
INC[carry][shift][#]□ACs, ACd [,skip]

Increment. $AC_S + 1$ → AC_d; affects Carry and Overflow flags.



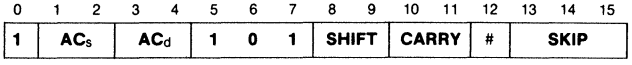
ADC[carry][shift][#]□ACs, ACd [,skip]

Add Complement. $\overline{AC_S} + AC_d$ → AC_d; affects Carry and Overflow.



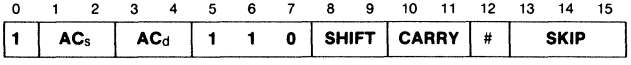
SUB[carry][shift][#]□ACs, ACd [,skip]

Subtract. AC_d — AC_S → AC_d; affects Carry and Overflow.



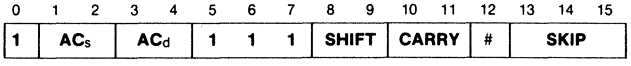
ADD[carry][shift][#]□ACs, ACd [,skip]

Add. $AC_S + AC_d$ → AC_d; affects Carry and Overflow.



AND[carry][shift][#]□ACs, ACd [,skip]

And. $AC_S \wedge AC_d$ → AC_d.



Base Carry Values

Mnemonic	Bits 10	11	Carry Value Used as Base
Omitted	0	0	Current Carry
Z	0	1	Zero
O	1	0	One
C	1	1	Complement of current Carry

Load/No-Load Condition

Mnemonic	Bit 12	Operation
Omitted	0	Load result in destination accumulator
#	1	Do not load result

Shift Operation

Mnemonic	Bits 8	9	Function
Omitted	0	0	No shift
L	0	1	Left rotate
R	1	0	Right rotate
S	1	1	Swap bytes

Skip Condition Codes

Mnemonic	Bits 13	14	15	Skip Condition
Omitted	0	0	0	Do not skip
SKP	0	0	1	Always skip
SZC	0	1	0	Skip if zero Carry
SNC	0	1	1	Skip if non-zero Carry
SZR	1	0	0	Skip if zero result
SNR	1	0	1	Skip if non-zero result
SEZ	1	1	0	Skip if either Carry or result zero
SBN	1	1	1	Skip if both Carry and result non-zero

Note

= Load/No-Load bit.

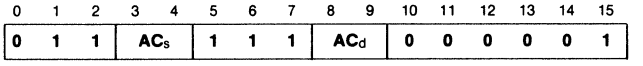
□ = Required separator.

A No-Load-No Skip instruction is interpreted as a trap if the Trap Enable flip-flop is set.

Arithmetic and Logic Instructions (Continued)

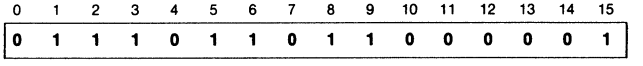
OR□ACs, ACd

Or. ACs V ACd → ACd.



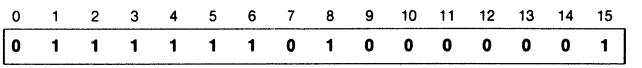
MUL

Unsigned Multiply. AC0 + (AC1 x AC2) → AC0, AC1.



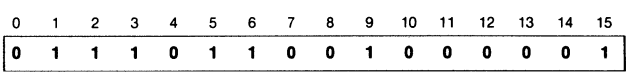
MULS

Signed Multiply. AC0 + (AC1 x AC2) → AC0, AC1.



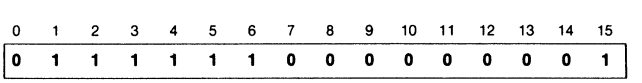
DIV

Unsigned Divide. (AC0, AC1)/AC2, quotient → AC1, remainder → AC0; Carry and Overflow = 1 if overflow occurs, Carry = 0 if not.



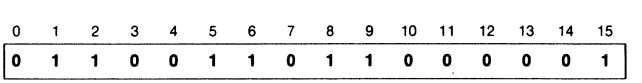
DIVS

Signed Divide. (AC0, AC1)/AC2, quotient → AC1, remainder → AC0; Carry and Overflow = 1 if overflow occurs, Carry = 0 if not.



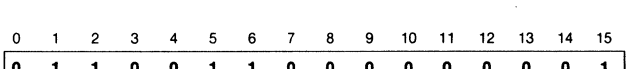
NORM

Normalize. Move the 32 bits in (AC0, AC1) to the left until high-order bit of AC0 = 1; number of steps required is subtracted from AC2; affects Overflow flag.



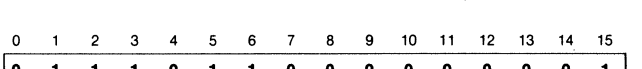
SLLD

Shift Logically Left. Shift the 32 bits in (AC0, AC1) logically left n times; zeroes shifted to LSB of AC1; n is contents of AC_2 ($1 \leq n \leq 31$).



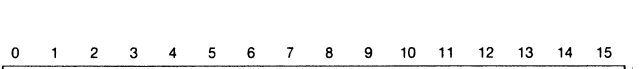
SALD

Shift Arithmetically Left. Shift the 32 bits in (AC0, AC1) to the left n times; zeroes shifted to LSB of AC1; set Overflow flag on first sign change; n is contents of AC2 ($1 \leq n \leq 31$).



SLRD

Shift Logically Right. Shift the 32 bits in (AC0, AC1) logically right n times; zeroes shifted to MSB of AC0; n is contents of AC2 ($1 \leq n \leq 31$).

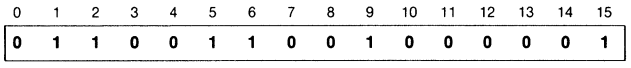


6

Arithmetic and Logic Instructions (Continued)

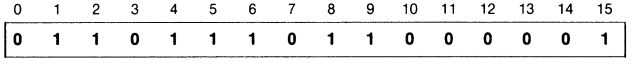
SARD

Shift Arithmetically Right. Shift the 32 bits in (AC0, AC1) arithmetically to the right n times; the MSB (sign) of AC0 is extended; n is contents of AC2 ($1 \leq n \leq 31$).



SKNV

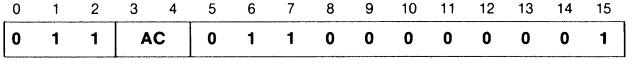
Skip on Not Overflow. Skip next instruction if Overflow = 0; then reset Overflow flag to 0.



Stack Instructions

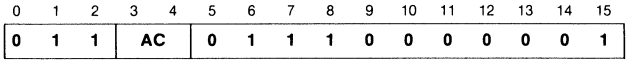
PSHA□AC

Push Accumulator. SP + 1 → SP, AC → (SP).



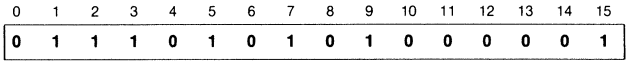
POPA□AC

Pop Accumulator. (SP) → AC, SP-1 → SP.



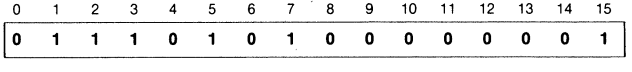
PSHF

Push Flags. SP + 1 → SP, PSW → (SP).



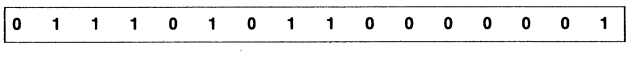
POPF

Pop Flags. (SP) → PSW, SP-1 → SP.



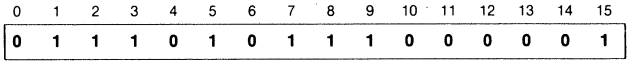
POPJ

Pop PC and Jump. (SP) → PC, SP-1 → SP.



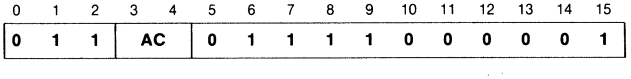
PSHR

Push Return Address. SP + 1 → SP, CA + 2 → (SP).



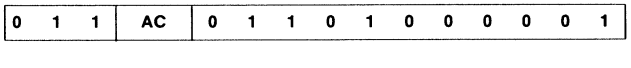
TOPR□AC

Read Top of Stack. (SP) → AC.



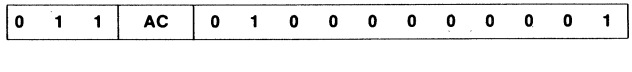
TOPW□AC

Write Top of Stack. AC → (SP).



MTSP□AC

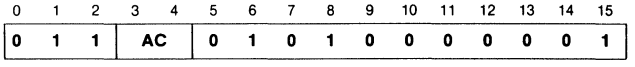
Move to Stack Pointer. AC → SP.



Stack Instructions (Continued)

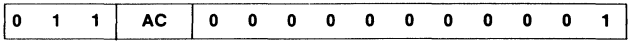
MFSP□AC

Move From Stack Pointer. SP → AC.



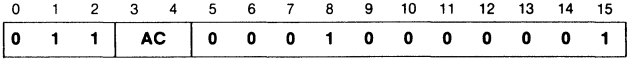
MTFP□AC

Move to Frame Pointer. AC → FP.



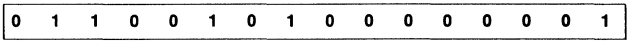
MFFP□AC

Move From Frame Pointer. FP → AC.



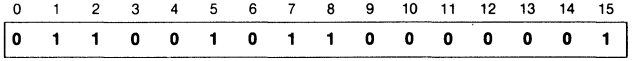
SAV

Save. Push the 5-word return block (AC0, AC1, AC2, FP |Carry*, $AC3_{1-15}$|) on stack, then load FP and AC3 with contents of SP.



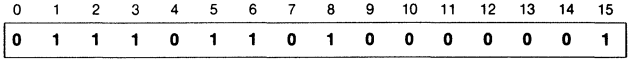
RET

Return. SP is loaded with contents of FP, then the 5-word return block is popped to (Carry*, PC_{1-15}), FP, AC3, AC2, AC1, and AC0, respectively.



DSP

Decrement Stack Pointer. SP-1 → SP.



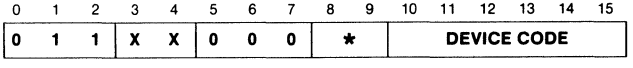
Notes

SP = Stack Pointer
FP = Frame Pointer
PSW = Program Status Word
CA = Current Address (PC-1)
*In 64K-word mode, Carry bit is not involved in SAV and RET and is replaced by $AC3_0$ and PC_0, respectively.

I/O Instructions

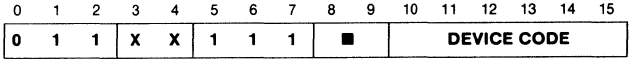
NIO[*]□device

No Data Transfer.



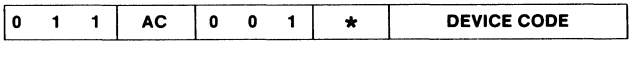
SKP■□ device

Skip on Busy/Done Flags. Skip next instruction if Busy/Done meets test condition.



DIA[*]□ AC, device

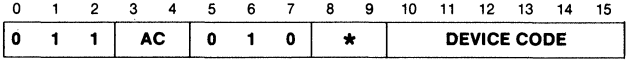
Data In From Register A. A → AC.



I/O Instructions (Continued)

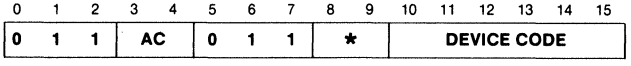
DOA[*]□ AC, device

Data Out to Register A. AC → A.



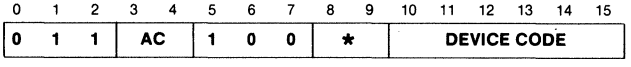
DIB[*]□ AC, device

Data In From Register B. B → AC.



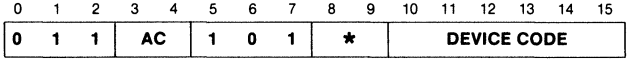
DOB[*]□ AC, device

Data Out to Register B. AC → B.



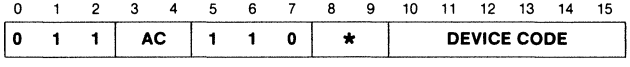
DIC[*]□ AC, device

Data In From Register C. C → AC.



DOC[*]□ AC, device

Data Out to Register C. AC → C.



Note

Device code = any number in the current radix (octal only for PEPBUG 45) between 0 and 77 octal except reserved codes 0 and 1, also may be the following standard mnemonics (or other user-defined mnemonics with the macro-assembler):

* Busy/Done Control Codes

Mnemonic	Bits 8	9	Operation
Omitted	0	0	Does not affect Busy and Done flags
S	0	1	START the device by setting Busy = 1 and Done = 0
C	1	0	CLEAR both Busy and Done to 0 and idle the device
P	1	1	PULSE the device. Its effect depends on device

Notes

X = Don't care.
* = Busy/done control code.
■ = Busy/done test code.
□ = Required separator.

■Busy/Done Test Codes

Mnemonic	Bits 8	9	Test Condition
BN	0	0	Busy is Non-Zero
BZ	0	1	Busy is Zero
DN	1	0	Done is Non-Zero
DZ	1	1	Done is Zero

Device Code Symbols

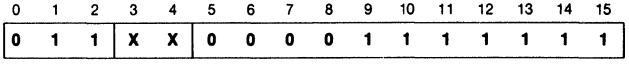
Mnemonic	Octal	Meaning
TTI	10	TTY input
TTO	11	TTY output
PTR	12	Reader
PTP	13	Punch
RTC	14	Real-time clock
LPT	17	Line printer
SMS	61	SMS disk drive
CPU	77	Console

Refer to "Control Instructions" regarding use of device code 77.

Control Instructions

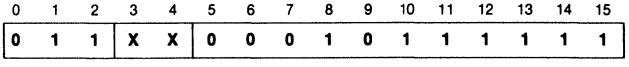
INTEN

Interrupt Enable. 1 → INTON; allows one more instruction to be executed before 1 → INTON.
Alternate assembler format:
NIOS□0, CPU



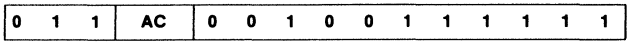
INTDS

Interrupt Disable. 0 → INTON.
Alternate assembler format:
NIOC□0, CPU



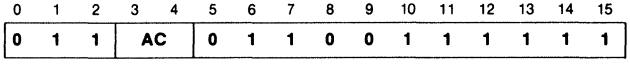
READS□AC

Read Console Switch Register. SW → AC.
Alternate assembler format:
DIA □AC, CPU



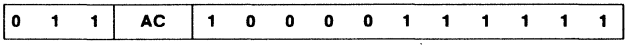
INTA□AC

Interrupt Acknowledge. The device code of the highest priority device requesting interrupt is loaded to bits 10-15 of AC.
Alternate assembler format:
DIB □AC, CPU



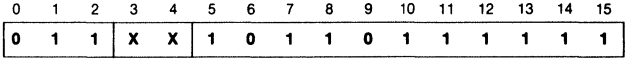
MSKO□AC

Mask Out. Enables specific devices to request interrupts.
Alternate assembler format:
DOB □AC, CPU



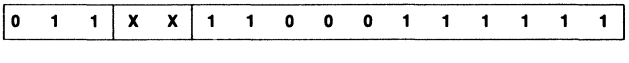
IORST

I/O Reset. Clear busy/done and interrupt enable flags of all I/O devices.
Alternate assembler format:
DIC C □0, CPU



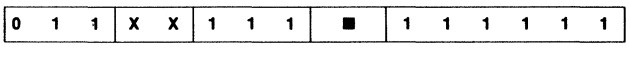
HALT

Halt the Processor. Only console operations are recognized.
Alternate assembler format:
DOC □0, CPU



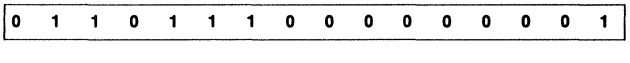
SKP■□ CPU

Skip on Interrupt-On Flag. Skip next instruction if the INTON flag fulfills the specified test conditions.



WAIT

Wait for Interrupt. Console, interrupt, and data channel request are recognized.



6

Control Instructions (Continued)

TRAP

Trap. CA → 46_8, (47_8) → PC.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	X	X	X	X	X	X	X	X	X	X	X	1	0	0	0

ETRP

Enable Trap Instruction. Default state by $\overline{MR}$.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	1	1	1	1	0	0	1	0	0	0	0	0	1

DTRP

Disable Trap Instruction.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1

E64K

Enable 64K-words mode.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	0	1	1	1	0	0	1	0	0	0	0	0	1

D64K

Disable 64K-words mode. Default state by $\overline{MR}$.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	0	1	1	1	0	1	0	0	0	0	0	0	1

Most of the Control instructions are a subset of I/O instructions using device code 77 octal ("all 1s," mnemonic: CPU); in device code 77 instructions, the Busy/Done control affects the Interrupt-On flag instead of the Busy and Done flags. Use of the Control mnemonics sets the Interrupt-On flag (via bits 8 and 9) as shown; however, with READS, INTA, MSKO, IORST or HALT, the alternate I/O mnemonics, which are shown for each instruction, may be used to control the Interrupt-On flag according to the "Interrupt-On Control Codes" table.

★ Interrupt-On (INTON) Control Codes

Mnemonic	Bits 8	Bits 9	Operation
Omitted	0	0	No effect
S	0	1	Set INTON = 1
C	1	0	Reset INTON = 0
P	1	1	No effect

■ Interrupt-On (INTON) Test Codes

Mnemonic	Bits 8	Bits 9	Test Condition
BN	0	0	Skip on INTON = 1
BZ	0	1	Skip on INTON = 0
DN	1	0	Reserved
DZ	1	1	Reserved

Notes
X = Don't care.
★ = Interrupt-On Control.
■ = Interrupt-On Test.
□ = Required separator.

FAIRCHILD
A Schlumberger Company

F9414
4-Chip Data Encryption Set

Microprocessor Product

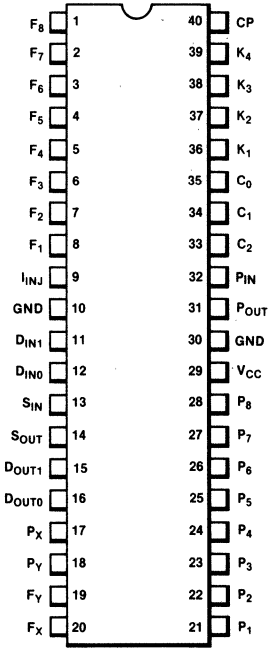
Description

The Fairchild F9414 4-Chip Data Encryption Set consists of four similar 40-pin I^3L® LSI devices (the 9414-1, 9414-2, 9414-3, and 9414-4), and is designed to implement the National Bureau of Standards data encryption standard (DES) algorithm (FIPS-46). The set uses a 56-bit key word to encipher or decipher a 64-bit word that is stored in 8 bytes; 2 bits of each byte are distributed to each of the four chips.

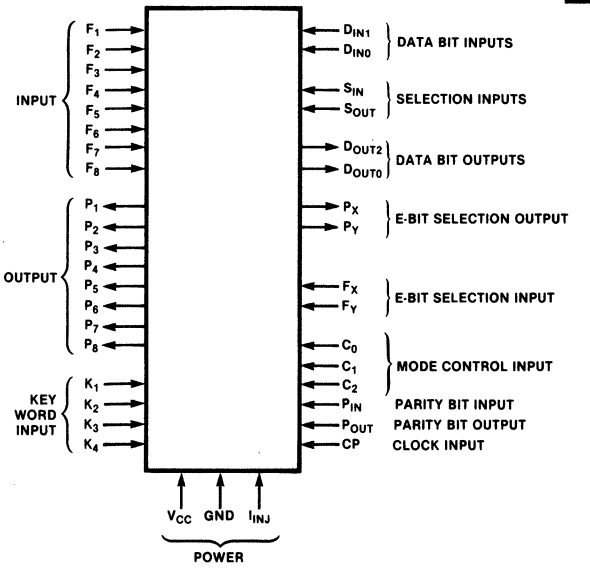
The major elements of each chip include a pair of data registers, four 8-bit shift (key) registers, control logic, and two 64-word by 4-bit read-only memories (ROMs). The F9414 encryption set has passed the NBS functional validation test.

- **High Throughput**
- **LSTTL Input/Output**
- **Single Clock**
- **Parity Testing**
- **Simultaneous Load and Output Data**
- **Cipher Feedback and Block Chaining**
- **3-State Data Buffers**
- **Single 5V Power Supply**
- **5 MHz Operation Typical**
- **Data Throughput — 4.8 μs Per 64-Bit Word**

Connection Diagram



Signal Functions



6

Signal Descriptions

The F9414 input and output signals are described in table 1.

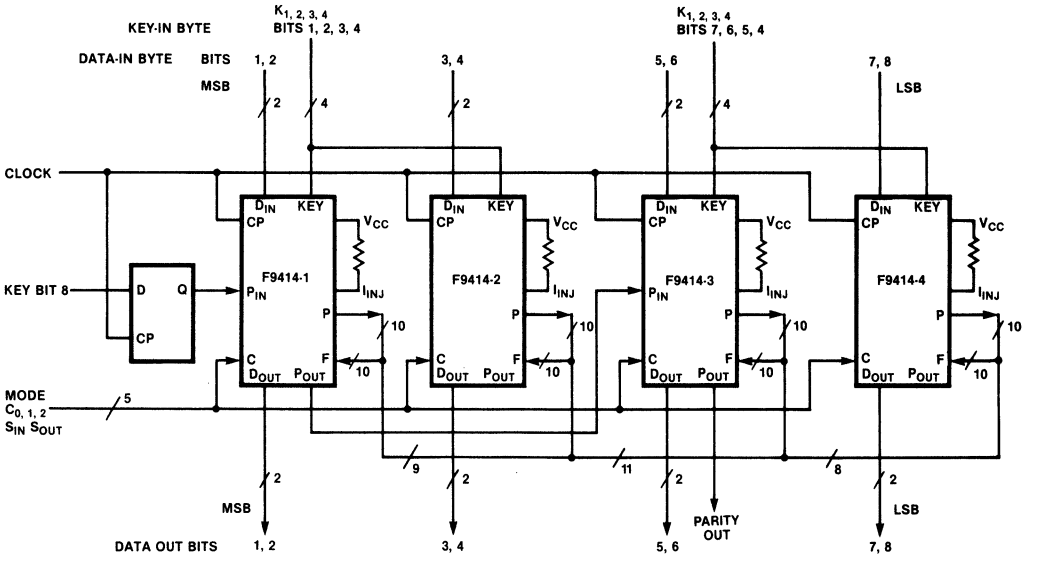
Functional Description

The set operates with a 56-bit key word to encipher or decipher a 64-bit data word that is stored in 8 bytes; 2 bits of each byte are distributed to each of the four chips (see figure 1). The key consists of 64 bits in 8 bytes; bit 8 of each byte is parity. Bits 1 through 4 go to both chip 1 and 2; bits 4 through 7 go to chips 3 and 4. The four chips together also store the 64-bit plaintext or ciphertext word. The chips have separate data inputs and outputs, so the block of data to be processed can be input as the previous block is being output. This overlap permits the processing of a 64-bit block in 24 clock pulses at a 5 MHz typical clock frequency. This results in data throughput of 13.3 MHz (75 ns) per bit, or 200 kHz (4.8 μs) per 64-bit word.

Table 1 F9414 Signal Descriptions

Mnemonic	Pin No.	Name	Description
F_1 - F_8	8, 7, 6, 5, 4, 3, 2, 1	Interconnect Lines	Input signals; interconnect with P_1-P_8 to implement the permutation function, P, of the algorithm.
P_1 - P_8	21 - 28	Interconnect Lines	Output signals; interconnect with F_1-F_8 to implement the permutation function, P, of the algorithm.
K_1 - K_4	36 - 39	Keyword	Input signal for 4 bits of the keyword.
D_{IN0}, D_{IN1}	12, 11	Data In	Data inputs for 2 bits of the data word.
S_{IN}	13	Select In	Input signal that selects the exclusive-OR function.
S_{OUT}	14	Select Out	Input signal that selects the output function.
D_{OUT0}, D_{OUT1}	16, 15	Data Out	Output lines for the data bits.
P_X, P_Y	17, 18	E-Bit Select	Output signal; E-bit selection for interconnection with F_X, F_Y.
F_X, F_Y	20, 19	E-Bit Select	Input signal; E-bit selection for interconnection with P_X, P_Y.
C_0, C_1, C_2	35, 34, 33	Control	Input signals used to control the F9414 in one of five modes.
P_{IN}	32	Parity In	Parity bit input signal
P_{OUT}	31	Parity Out	Parity bit output signal
CP	40	Clock	Input signal
V_{CC}	29	Power	+5 V ±5% power supply
I_{INJ}	9	Power	Injection current input
GND	10	Ground	0 V reference.

Figure 1 4-Chip Encryption Set



The key register is capable of hold, left shift (encipher), or right shift (decipher) operations, by one or two positions, as required by each of the 16 rounds of the algorithm (see figure 2). Each device also includes logic for the control of these registers during load and cipher operations. The 64-bit word by 4-bit ROMs in each device implement the S-boxes of the algorithm.

The major differences among the four devices are the masking of the ROM codes and the key bits that are selected as ROM addresses, according to the E-bit selection table of the algorithm.

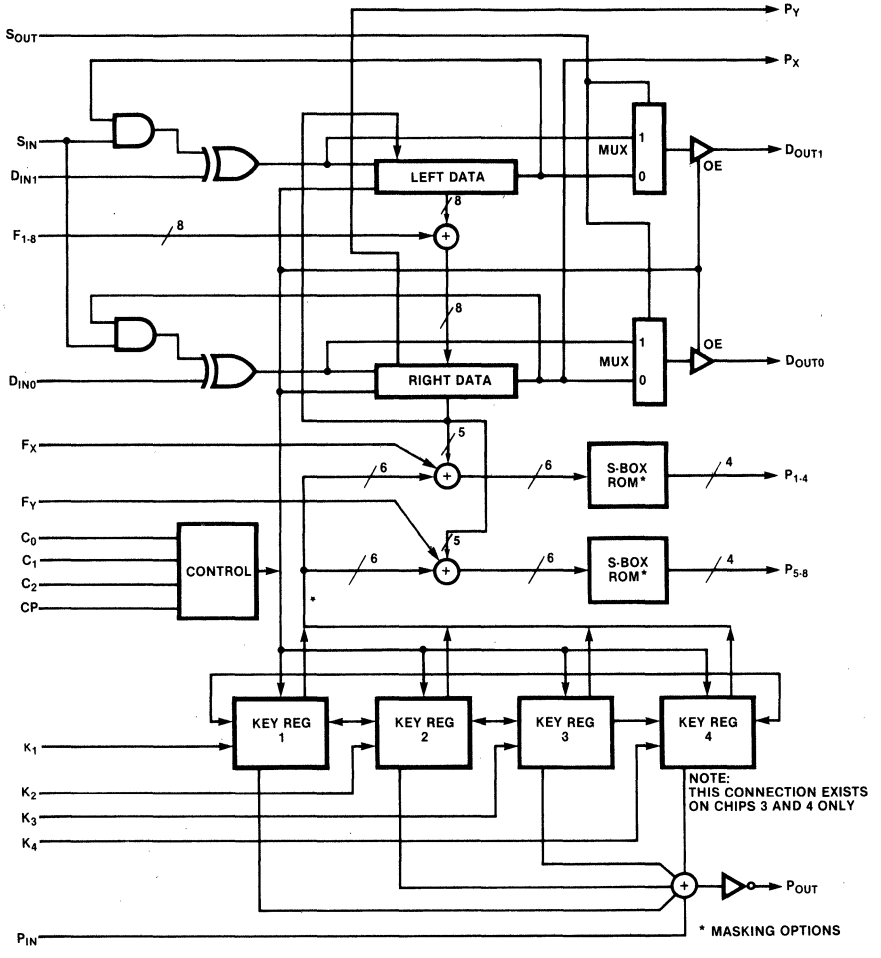
A set of eight output signals ($P_{1\text{-}8}$) and input signals ($F_{1\text{-}8}$) is interconnected between chips to implement the permutation function, P, of the algorithm. An additional set of outputs (P_X and P_Y) and inputs (F_X and F_Y) is used to interconnect the chips as required by columns 1 and 6 of the E bit-selection table.

Implementation of the Algorithm

Initial permutation is accomplished in the F9414 chip set by the manner in which the data is loaded. The D_{IN0} input of chip 1 loads bit 1 of each byte, D_{IN1} of chip 1 loads bit 2 of each byte, D_{IN0} of chip 2 loads bit 3 of each byte, etc. After eight clock cycles, the four registers receiving data bits 2, 4, 6, and 8 of each input byte comprise the L_0 block of 32 bits in permuted order within the four devices. The four registers receiving bits 1, 3, 5, and 7 of each byte hold the R_0 block. Therefore, each chip slice contains one byte each of the L_0 and R_0 blocks.

Further shifting of the bits and extracting outputs from the right end of each byte implements the inverse permutation, $1P^{-1}$. Each column of the inverse permutation may be found in a register byte, and the first 8 bits (40, 8, 48, etc.) required by row 1 of the inverse permutation table are at the output ends of the shift registers.

Figure 2 F9414 Block Diagram (One Unit)



The 28 key bits in the top half, C_0, of the key permutation function are duplicated in the key registers of F9414-1 and F9414-2, while key bits in the bottom half, D_0, occupy the registers of both the F9414-3 and F9414-4. In each device, key register 4 holds the last 4 bits of both halves of the key permutation function. Each of the 16 iterations involves a left rotation (encipher) or right rotation (decipher) of the key registers.

During the key shift schedule, chips 1 and 2 bypass the right half of key register 4, and chips 3 and 4 bypass the left. This results in the key alignment returning to its original position after a total of 28 shifts from the 16 alterations.

An internal 1-bit right realignment is required by a change from encipher to decipher, after the key has been entered. This, and the reverse (left realignment for decipher to encipher), are performed by the F9414 control logic, which must be stable prior to the loading of the last data byte. When clocked at the same time as a load-key code, the data registers all fill with logic ones.

The results of the exclusive-OR of the key bits and data words derived from R_0 in the calculation of f(R,K) are taken, 6 bits at a time, to address a set of eight 64 × 4 S ROMs (i.e., S boxes). Two S ROMs per chip, each with four output bits, provide the 32 bits that are then permuted per primitive function P, by chip-to-chip interconnection. The effective result of the interconnect is exclusive-ORed with the L_0 block and the entire algorithm is repeated 16 times.

The F9414 is structurally designed for high throughput. Since no I/O ports are used for both entering data and reading results, a potential bottleneck is avoided. The 64-bit data word is entered into the F9414 data registers 1 byte at a time at the D_0,D_1 inputs. The MSB of data goes to D_0 of the F9414-1. The result is output 1 byte at a time on the Q_0,Q_1 pins, MSB output first. Similarly, the keyword is entered 1 byte at a time at its own dedicated inputs (K_1-K_4). Table 2 shows the distribution of the keyword to the four F9414 devices.

Table 2 Keyword Distribution

Keyword	F9414-1 Key Reg.	F9414-2 Key Reg.	F9414-3 Key Reg.	F9414-4 Key Reg.
8 MSB	1	1		
7	2	2		
6	3	3		
5	4	4	4	4
4			3	3
3			2	2
2 LSB			1	1
1 Parity (Option)	P_{IN}			

The keyword is 56 bits long but, if desired, an optional parity bit can be included with each byte of key, making the keyword 64 bits long. Parity does not in any way affect the encryption or decryption, and is taken across the keyword register, not across the K_1-K_4 inputs. Parity across 1 byte of keyword is taken by passing the parity bit of the keyword through a delay flip-flop to P_{IN} of the F9414-1 or F9414-2, and through P_{OUT} of the F9414-1 or F9414-2 into P_{IN} of the F9414-3 or F9414-4. The final parity sum is available on P_{OUT} of the F9414-3 or F9414-4.

The functions of the F9414 (load key, load data, encryption/decryption, and wait) are controlled by the C_0-C_2 inputs. Data and key are clocked in and/or out on low-to-high clock transitions. Loading a key sets the data registers to all high.

The F9414 enables simultaneous input and output of data; i.e., the results of a DES cipher operation can be clocked out on the same low-to-high transition that loads the next word to be processed. Thus, a complete input and output cycle (LOAD/READ DATA) takes just eight clocks. Since the algorithm requires 16 clocks, an entire DES iteration can be accomplished in 24 clocks. At a typical clock frequency of 6 MHz, this translates into a 16 MHz bit rate, a very fast LSI implementation of the DES. This high throughput ensures that the F9414 set is capable of keeping pace with practically every application, and this speed is available over the full military temperature range.

Implementation of Cipher Feedback

In cipher feedback (see figure 3), the present 64-bit data input is exclusive-ORed with the output of the encryption unit, and the result of this operation is transmitted and also fed back into the encryption unit to perpetuate the feedback. At the receiver, the received 64-bit vector is first exclusive-ORed and then deciphered.

Figure 4 illustrates the cipher feedback (CFB) transmitter operation. A 64-bit buffer is necessary for storing the input word external to the F9414 and can be provided with two F9423 first-in first-out (FIFO) buffer memories. Both receiver and transmitter operate in the same mode and start with the same (arbitrary) initialization word in the buffer. If the initialization is not done, the first 64 bits of data at the receiver are erroneously deciphered.

To encrypt 1 byte of data, one iteration of the DES algorithm is performed on the contents of the buffer. Then the MSB output from the F9414 is exclusive-ORed with the data byte and the result is transmitted. Additionally, the result of the exclusive-ORing is shifted into the least significant position of the buffer, while all other bytes are shifted and the former MSB discarded. This causes all following encryptions to depend on the present transmission, providing greater security than when each encryption depends only on the present data byte.

At the receiver (see figure 5), the transmission is shifted into the least significant position of the buffer and one DES iteration is performed. Since the receiver has used the same data word as the transmitter, this generates the same exclusive-OR mask as was used at the transmitter. Therefore, exclusive-ORing the next received byte with the MSB of the F9414 output recovers the data byte.

The transmitter and receiver must be operating in synchronization in cipher feedback. If synchronization is lost or an erroneous bit received, 64 bits of data will be incorrectly deciphered.

Figure 3 Cipher Feedback Implementation

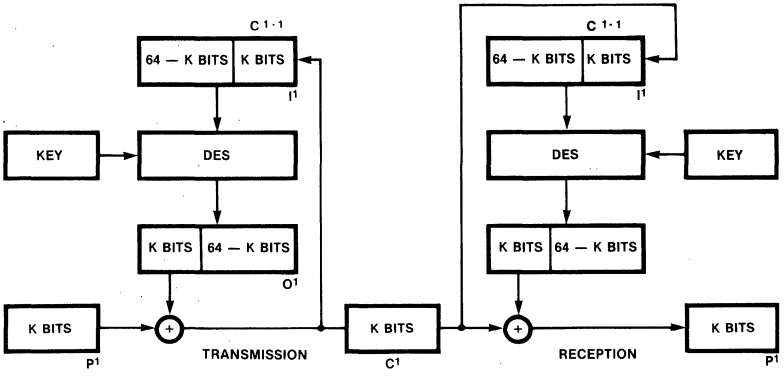
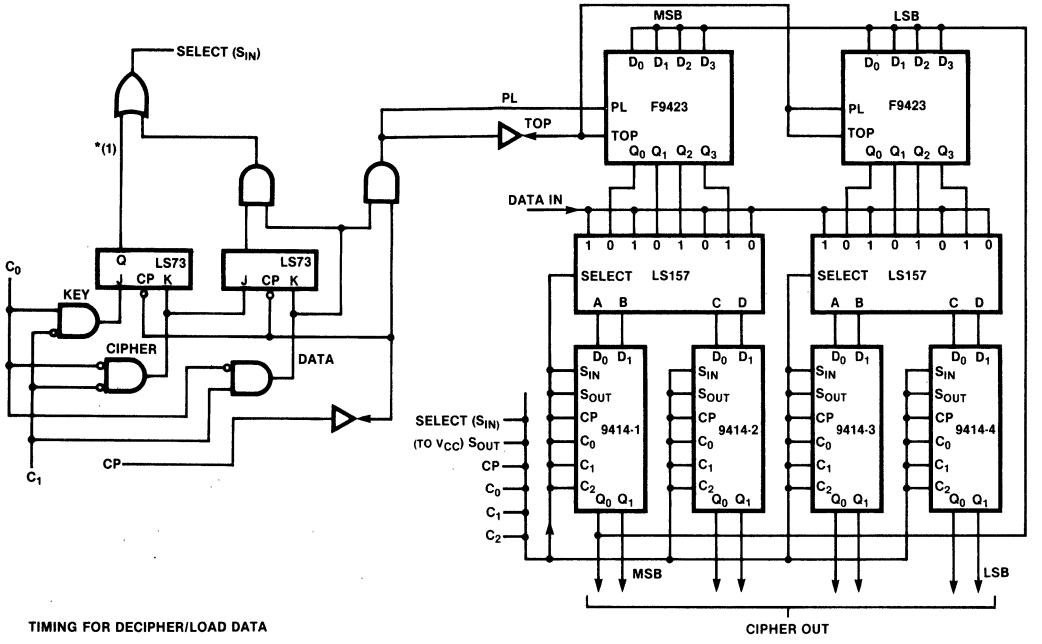
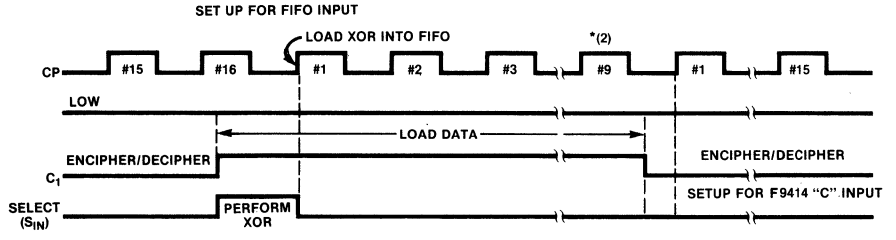


Figure 4 Cipher Feedback Transmitter

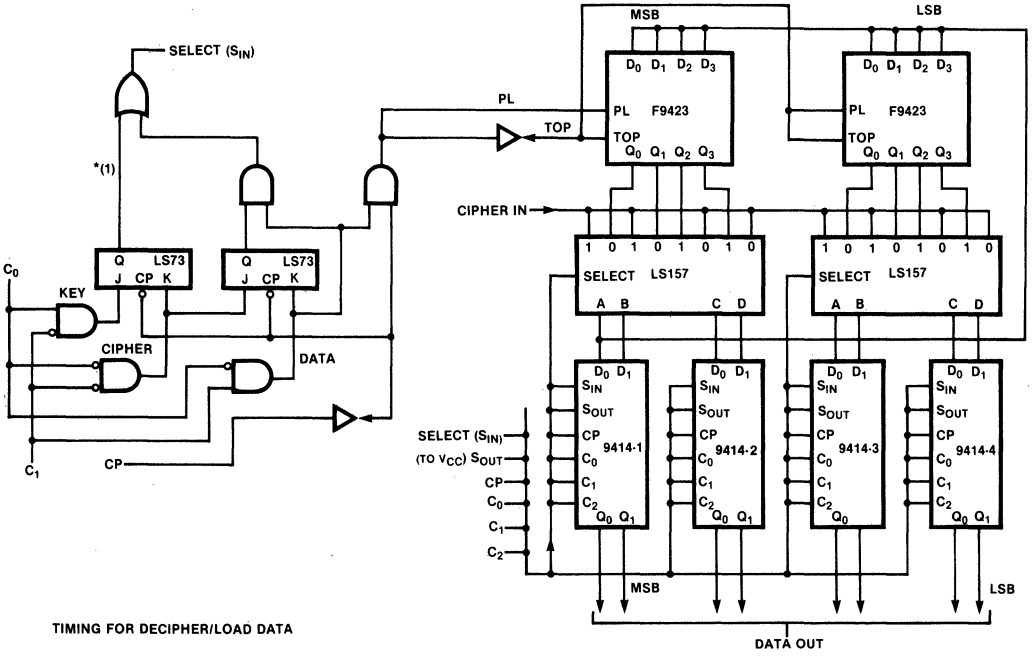


TIMING FOR DECIPHER/LOAD DATA

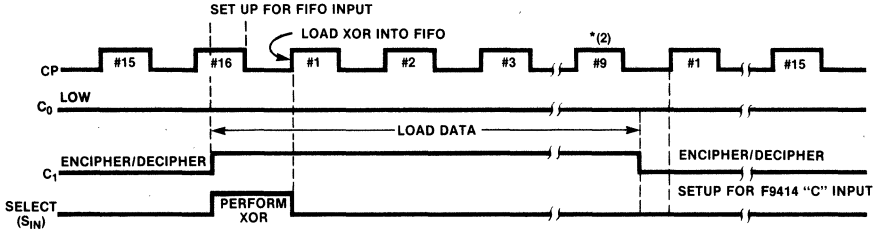


NOTES:
1. 8 byte initialization vector following LD KEY
2. 9 clock pulses required to complete load data operation

Figure 5 Cipher Feedback Receiver



TIMING FOR DECIPHER/LOAD DATA



NOTES:
1. 8 byte initialization vector following LD KEY
2. 9 clock pulses required to complete load data operation

Implementation of Cipher Block Chaining

Cipher block chaining (see figure 6) is similar to cipher feedback in that successive transmissions are made dependent on previous transmissions, thereby increasing the level of security. The cipher block chaining transmitter takes the present 64-bit input vector and exclusive-ORs it with the output of the encryption unit, then performs an encryption on the result. The result of the encryption is transmitted and also exclusive-ORed with the next 64-bit vector, continuing the chaining process. The receiver runs synchronously with the transmitter, and recovers the data by performing a decryption and then an exclusive-OR on the received 64 bits.

Receiver and transmitter must operate in different modes: encrypt and decrypt (see figures 7 and 8). No data buffering is necessary at the transmitter, but the receiver needs a 64-bit buffer to store the previous transmission. Both receiver and transmitter must start with the same initialization data or the first 64 bits of transmission will be incorrectly deciphered.

Internal exclusive-OR gates on the F9414 make implementation of the cipher block chaining transmitter especially simple. When S_{IN} is high, the exclusive-OR of the D inputs and Q outputs is input to the F9414 register. Since the F9414 can input and output simultaneously, the input data and the F9414 output are exclusive-ORed while the result of the DES iteration is being clocked out at the Q outputs. Therefore, no additional packages are required.

Figure 6 Cipher Block Chaining Mode with Terminal Block Padding

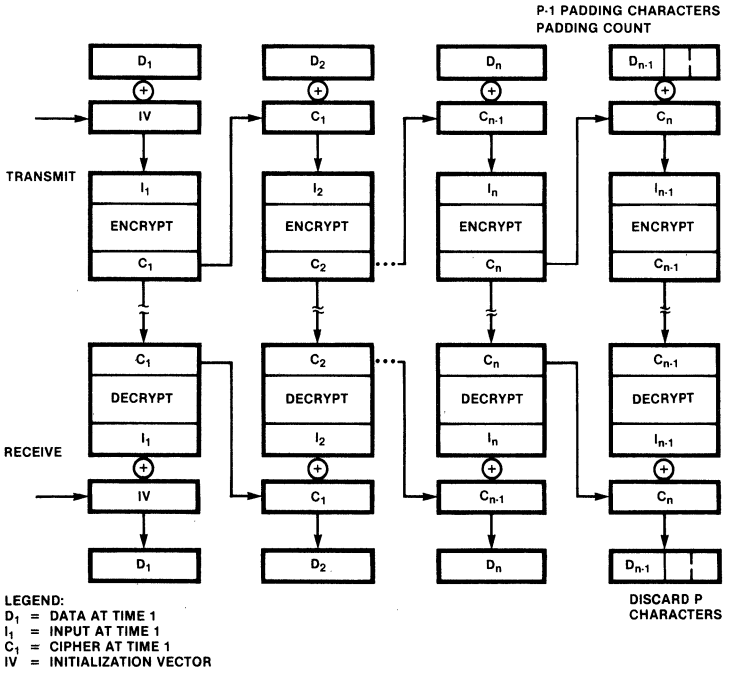
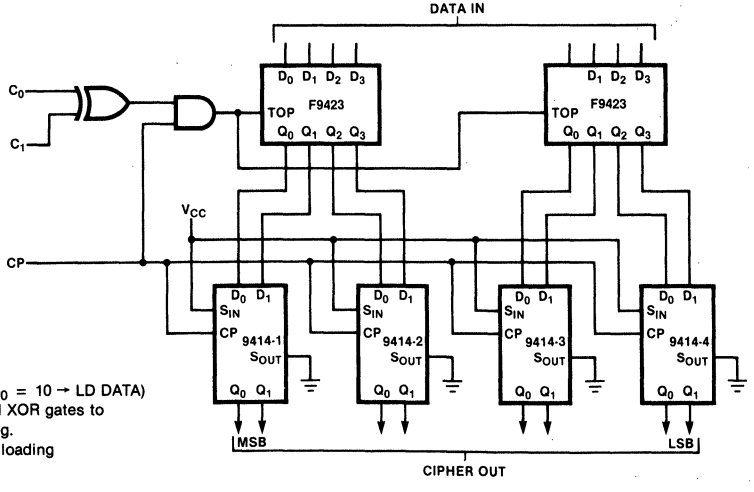
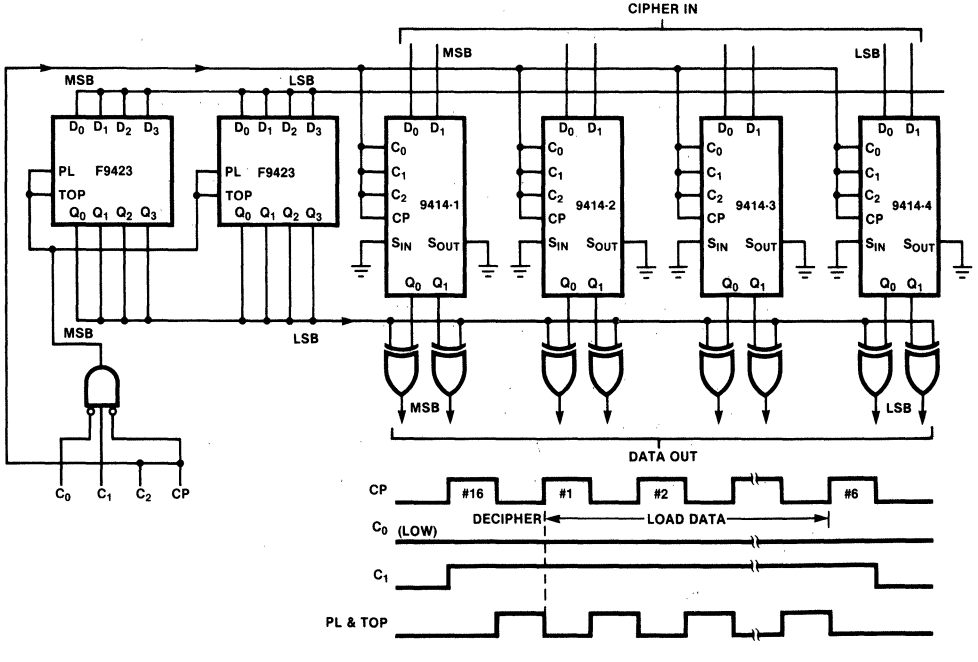


Figure 7 Cipher Block Chaining Transmitter



NOTES:
1. A high on S_{IN} (C_1 C_0 = 10 → LD DATA) enables the internal XOR gates to perform the chaining.
2. Hold S_{IN} low while loading initialization vector.

Figure 8 Cipher Block Chaining Receiver



Timing Characteristics

Signal timing diagrams for the data encryption set are shown in figures 9 through 11, and the timing characteristics are provided in table 3. The ac characteristics are: V_{CC} = 5V ±5%; T_A = 0°C to 70°C; C_L = 15 pF; and I_{INJ} = 85 to 125 mA.

Table 3 Timing Characteristics

Symbol	Parameter	Limits			Units	Comments
		Min	Typ	Max		
T_P	Prop. Delay, CP to P_{1-8}			155	ns	
T_P	Prop. Delay, CP to P_X, P_Y			110	ns	
T_P	Prop. Delay, CP to D_{OUT}			120	ns	S_{OUT} Low
T_P	Prop. Delay, CP to D_{OUT}		132	—	ns	S_{IN}, S_{OUT}, High
T_P	Prop. Delay, CP to P_{OUT} (9414-1, -2)			130	ns	C_{210} = XLH
T_P	Prop. Delay, CP to P_{OUT} (9414-3, -4)			145	ns	C_{210} = XLH
T_P	Prop. Delay, S_{IN} to D_{OUT}		75	—	ns	S_{OUT} High
T_P	Prop. Delay, S_{OUT} to D_{OUT}			85	ns	
T_P	Prop. Delay, D_{IN} to D_{OUT}		55	—	ns	S_{IN}, S_{OUT} High
T_P	Prop. Delay, C_{210} to D_{OUT}			105	ns	
T_P	Prop. Delay, P_{IN} to P_{OUT}			60	ns	
T_P	Prop. Delay, F_X, F_Y to P_{1-8}			100	ns	
T_S	Set-up Time, F_{1-8} to CP	50			ns	
T_S	Set-up Time, D_{IN} to CP	45			ns	C_{210} = XHL
T_S	Set-up Time, S_{IN} to CP	70			ns	C_{210} = XHL
T_S	Set-up Time, C_{210} to CP	110			ns	
T_S	Set-up Time, K_{1-4} to CP	50			ns	C_{210} = XLH
T_H	Hold Time, CP to F_{1-8}	5			ns	
T_H	Hold Time, CP to D_{IN}	0			ns	C_{210} = XHL
T_H	Hold Time, CP to S_{IN}	0			ns	C_{210} = XHL
T_H	Hold Time, CP to C_{210}	0			ns	
T_H	Hold Time, CP to K_{1-4}	10			ns	C_{210} = XLH
T_{PWH}	CP Pulse Width High	50			ns	

Figure 9 Load Key Timing Diagram

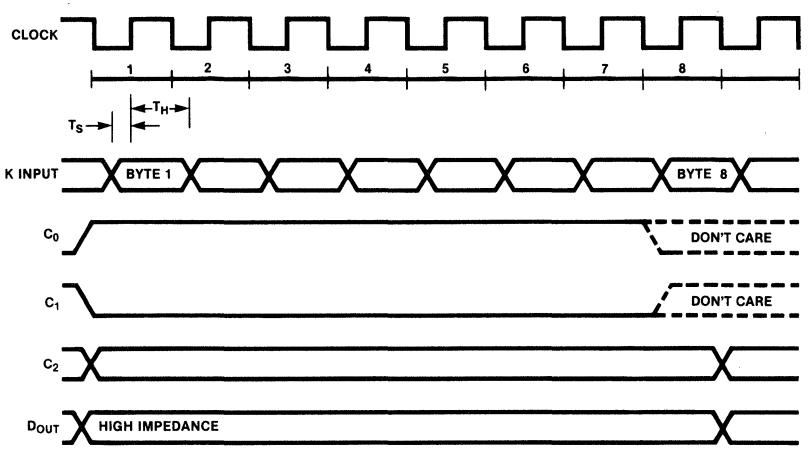


Figure 10 Load/Read Data Timing Diagram

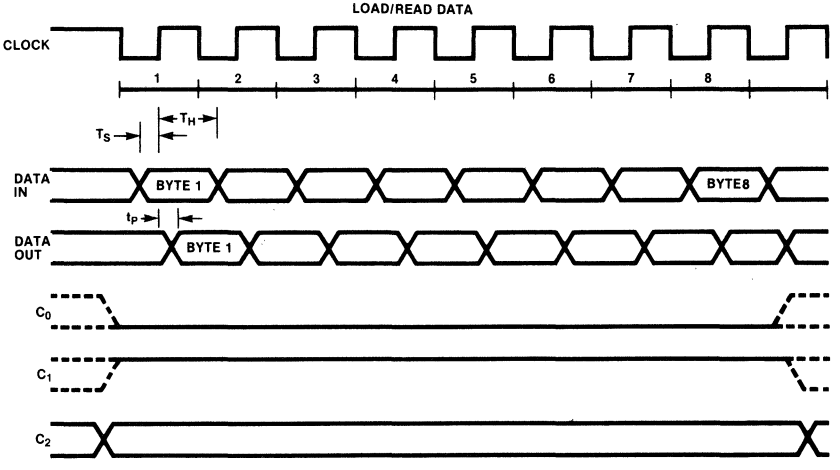
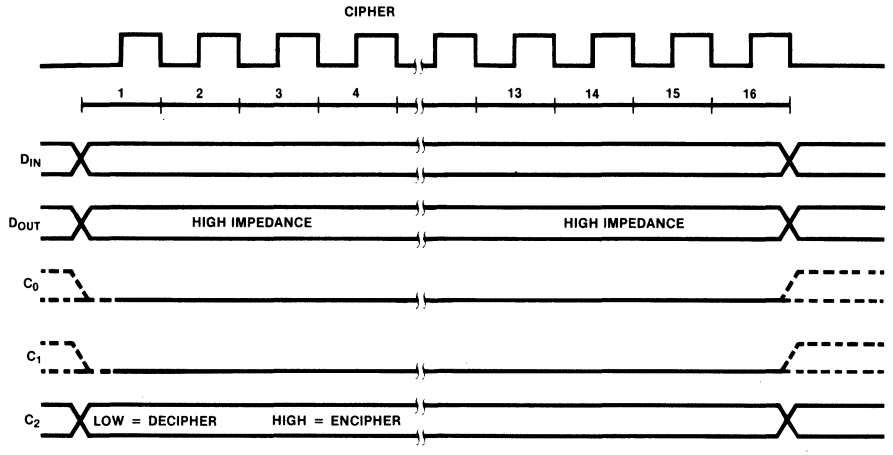


Figure 11 Cipher Timing Diagram



6

DC Characteristics

The dc characteristics of the data encryption set are provided in table 4. The dc characteristics are specified over operating temperature range, unless otherwise noted:

0 °C to 70 °C; $I_{INJ(min.)}$ = 85 mA; $I_{INJ(max.)}$ = 125 mA;
$V_{CC(min.)}$ = 4.75 V; $V_{CC(max.)}$ = 5.25 V.
Typical limits are at V_{CC} = 5.0 V, T_A = 25 °C.

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Storage Temperature	− 65 °, + 150 °C
Ambient Temperature under Bias	− 55 °, + 125 °C
V_{CC} Pin Potential to Ground Pin	− 0.5, + 6.0 V
Input Voltage (DC)	− 0.5, + 5.5 V
Input Current (DC)	− 20, + 5 mA
Output Voltage (Output High)	− 0.5, + 5.5 V
Output Current (DC) (Output Low)	+ 20 mA
Injector Current (I_{INJ})	+ 200 mA
Injector Voltage (V_{INJ})	− 0.5, + 1.8 V

Control Codes

Table 5 provides the control codes for the data encryption set.

Table 5 Control Codes

C_2	C_1	C_0		Clock Cycles
0	0	0	DECIPHER	16
1	0	0	ENCIPHER	16
X	0	1	LOAD KEY	8
X	1	0	LOAD DATA/OUTPUT DATA	8
X	1	1	WAIT	X

Table 4 DC Characteristics

Symbol	Parameter	Limits			Units	Test Conditions I_{INJ} = 100 mA
		Min	Typ	Max		
V_{IH}	Input High Voltage	2.0			V	Guaranteed Input High Voltage
V_{IL}	Input Low Voltage			0.8	V	Guaranteed Input Low Voltage
V_{CD}	Input Clamp Diode Voltage		−0.9	−1.5	V	V_{CC} = Min, I_{IN} = −18 mA
V_{OH}	Output High Voltage	2.4	3.4		V	V_{CC} = Min I_{OH} = −1.0 mA (D_{OUT}) I_{OH} = −400 μA (Other Outputs)
V_{OL}	Output Low Voltage		0.25	0.5	V	V_{CC} = Min, I_{OL} = 8.0 mA
	Input High Current, All Except CP		1.0	20	μA	V_{CC} = Max, V_{IN} = 2.7 V
I_{IH}	Input High Current, CP		1.0	40	μA	V_{CC} = Max, V_{IN} = 2.7 V
	Input High Current, All Inputs			1.0	mA	V_{CC} = Max, V_{IN} = 5.5 V
I_{IL}	Input Low Current, All Except CP Input Low Current, CP		−0.21 −0.42	−0.36 −0.72	mA	V_{CC} = Max, V_{IN} = 0.4 V
I_{OZH} I_{OZL}	Off State (High Impedance) Output Current, D_{OUT}			100 −100	μA μA	V_{CC} = Max, V_{OUT} = 2.4 V V_{CC} = Max, V_{OUT} = 0.5 V
I_{OS}	Output Short Circuit Current	−15		−100	mA	V_{CC} = Max, V_{OUT} = 0
I_{CC}	Supply Current		150	220	mA	V_{CC} = Max
V_{INJ}	Injector Voltage	1.0	1.3	1.5	V	I_{INJ} = 100 mA, V_{CC} = 5.0 V

Device Interconnection

Table 6 gives the interconnection information for the four-chip set.

Table 6 Device Interconnection

nF (f) to nP (p)*							
1F1 to 2P8		2F1 to 1P1		3F1 to 1P2		4F1 to 3P3	
1F2	1P7	2F2	2P7	3F2	1P8	4F2	2P5
1F3	3P4	2F3	3P7	3F3	3P8	4F3	4P6
1F4	3P5	2F4	4P2	3F4	2P6	4F4	1P6
1F5	4P5	2F5	1P5	3F5	4P8	4F5	3P6
1F6	2P4	2F6	3P2	3F6	4P3	4F6	2P3
1F7	4P4	2F7	4P7	3F7	1P3	4F7	1P4
1F8	3P1	2F8	2P2	3F8	2P1	4F8	4P1
1FX	4PX	2FX	1PX	3FX	2PX	4FX	3PX
1FY	2PY	2FY	3PY	3FY	4PY	4FY	1PY

*n indicates chip option
f and p indicate specific member

Ordering Information

Part Number	Package	Temperature Range
F9414 ST DC	Ceramic DIP	0°C to 70°C

Export Control

Cryptographic devices and technical data regarding them are subject to Federal Government export controls as specified in Title 22, Code Of Federal Regulations, Parts 121 through 128.

FAIRCHILD
A Schlumberger Company

F9423
FIFO Buffer Memory

Microprocessor Product

Description

The Fairchild F9423 is an expandable fall-through-type high-speed first-in, first-out (FIFO) buffer memory that is optimized for high-speed disk or tape controller and communication buffer applications. It is organized as 64 words by 4 bits and may be expanded to any number of words or any number of bits in multiples of four. Data may be entered or extracted asynchronously in serial or parallel, allowing economical implementation of buffer memories.

The F9423 has 3-state outputs that provide added versatility, and is fully compatible with all TTL families.

- **Serial or Parallel Input**
- **Serial or Parallel Output**
- **Expandable Without Additional Logic**
- **3-State Outputs**
- **Fully Compatible With All TTL Families**
- **Slim 24-Pin Package**

Device Organization

As shown in figure 1, the F9423 consists of three sections:

1. An input register with parallel and serial data inputs, as well as control inputs and outputs for input handshaking and expansion.
2. A 4-bit-wide, 62-word-deep fall-through stack with self-contained control logic.
3. An output register with parallel and serial data outputs, as well as control inputs and outputs for output handshaking and expansion.

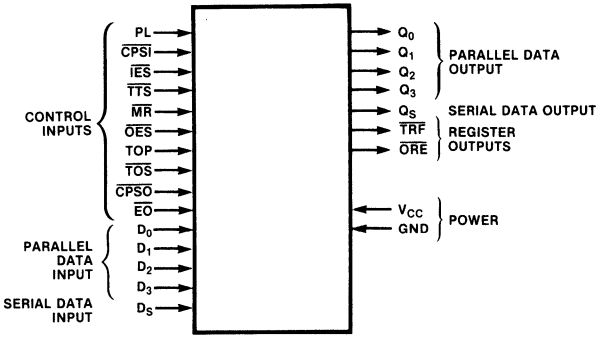
These three sections operate asynchronously and are virtually independent of one another.

Signal Functions

The F9423 FIFO signal functions are described in table 1

6

Signal Functions



Connection Diagram

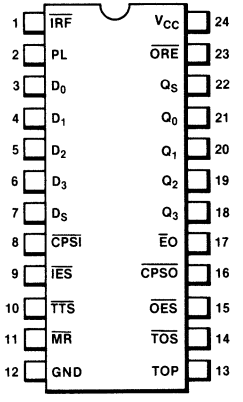
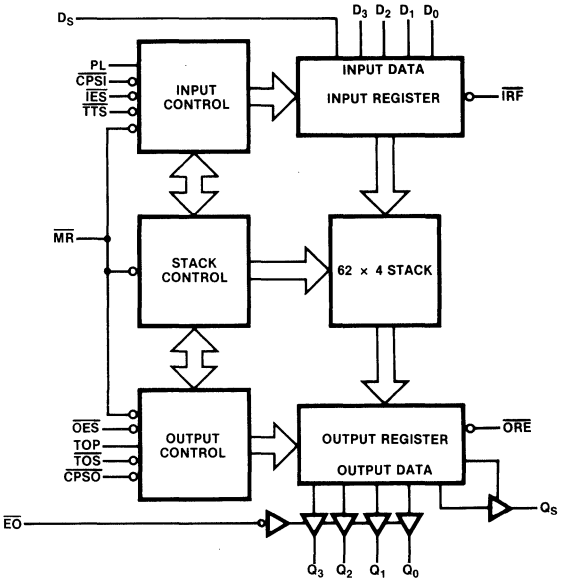


Table 1 Signal Functions

Mnemonic	Pin No.	Name	Description
Control Inputs			
PL	2	Parallel Load	Input signal that, when high, enables D_0 - D_3; not edge-triggered. Ones catching.
$\overline{CPSI}$	8	Serial Input Clock	Edge-triggered input signal that activates on the falling edge.
$\overline{IES}$	9	Serial Input Enable	Input signal that, when low, enables serial and parallel input.
$\overline{TTS}$	10	Transfer-to-Stack	Input signal that, when low, initiates fall-through.
$\overline{MR}$	11	Master Reset	Active-low input signal.
$\overline{OES}$	15	Serial Output Enable	Input signal that, when low, enables serial and parallel output.
TOP	13	Transfer Out Parallel	Input signal that, when high, enables a word to be transferred from the stack to the output register; not edge-triggered. (The $\overline{TOS}$ signal must be low for the transfer to occur.)
$\overline{TOS}$	14	Transfer Out Serial	Input signal that, when low, enables a word to be transferred from the stack to the output register; not edge-triggered. (The TOP signal must be high for the transfer to occur.)
$\overline{CPSO}$	16	Serial Output Clock	Edge-triggered input signal that activates on the falling edge.
$\overline{EO}$	17	Output Enable	Active-low input signal that allows data to be output.
Data Inputs			
D_0-D_3	3 - 6	Parallel Data	Parallel data inputs
D_S	7	Serial Data	Serial data inputs
Data Outputs			
Q_0-Q_3	18 - 21	Parallel Data	Parallel data outputs
Q_S	22	Serial Data	Serial data output
Register Status			
$\overline{IRF}$	1	Input Register Full	Output signal that, when low, indicates that the input register is full.
$\overline{ORE}$	23	Output Register Empty	Output signal that, when high, indicates that the output register contains valid data.
Power			
V_{CC}	24	Power Supply	Nominal +5 V
GND	12	Ground	Common power and signal return

Figure 1 F9423 Block Diagram



Functional Description

Input Register

The input register can receive data in either bit-serial or 4-bit parallel form. It stores this data until it is sent to the fall-through stack, and also generates the necessary status and control signals.

This 5-bit register (see figure 2) is initialized by setting flip-flop F3 and resetting the other flip-flops. The Q-output of the last flip-flop (FC) is output as the Input Register Full ($\overline{IRF}$) signal. After initialization, this output is high.

Parallel Entry

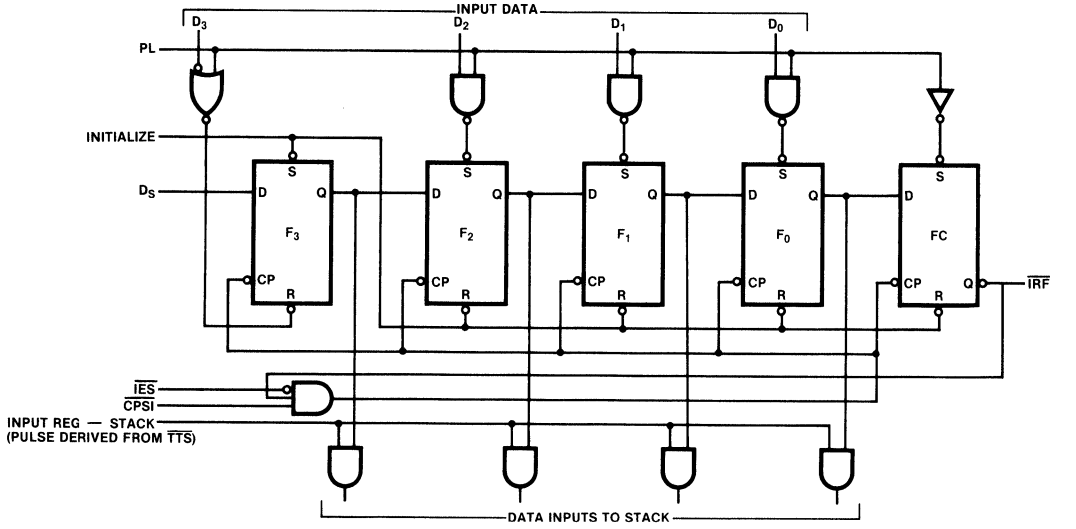
A high on the parallel load (PL) input loads the D_0 - D_3 inputs into the F_0 - F_3 flip-flops and sets the FC flip-flop. This forces the $\overline{IRF}$ output low, indicating that the input register is full. During parallel entry, the serial input clock ($\overline{CPSI}$) input must be low.

Serial Entry

Data on the serial data (D_S) input is serially entered into the shift register (F_3, F_2, F_1, F_0, FC) on each high-to-low transition of the $\overline{CPSI}$ input when the Serial Input Enable ($\overline{IES}$) signal is low. During serial entry, the PL input should be low.

After the fourth clock transition, the four data bits are located in flip-flops F_0 - F_3. The FC flip-flop is set, forcing the $\overline{IRF}$ output low and internally inhibiting $\overline{CPSI}$ pulses from effecting the register. Figure 3 illustrates the final positions in an F9423 resulting from a 256-bit serial bit train (B_0 is the first bit, B_{255} the last).

Figure 2 Input Register Conceptual Logic Diagram



Fall-Through Stack

The outputs of flip-flops F_0 - F_3 feed the stack. A low level on the transfer-to-stack ($\overline{TTS}$) input initiates a fall-through action; if the top location of the stack is empty, data is loaded into the stack and the input register is reinitialized. (Note that this initialization is delayed until PL is low.) Thus, automatic FIFO action is achieved by connecting the $\overline{IRF}$ output to the $\overline{TTS}$ input.

An RS-type flip-flop (the initialization flip-flop) in the control section records the fact that data has been transferred to the stack. This prevents multiple entry of the same word into the stack even though $\overline{IRF}$ and $\overline{TTS}$ may still be low; the initialization flip-flop is not cleared until PL goes low.

Once in the stack, data falls through automatically, pausing only when it is necessary to wait for an empty next location. In the F9423, the master reset ($\overline{MR}$) input only initializes the stack control section and does not clear the data.

Output Register

The output register (see figure 4) receives 4-bit data words from the bottom stack location, stores it, and outputs data on a 3-state, 4-bit parallel data bus or on a 3-state serial data bus. The output section generates and receives the necessary status and control signals.

Figure 3 Final Bit Positions Resulting from 256-Bit Serial Train

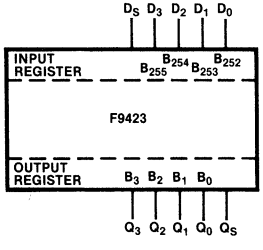
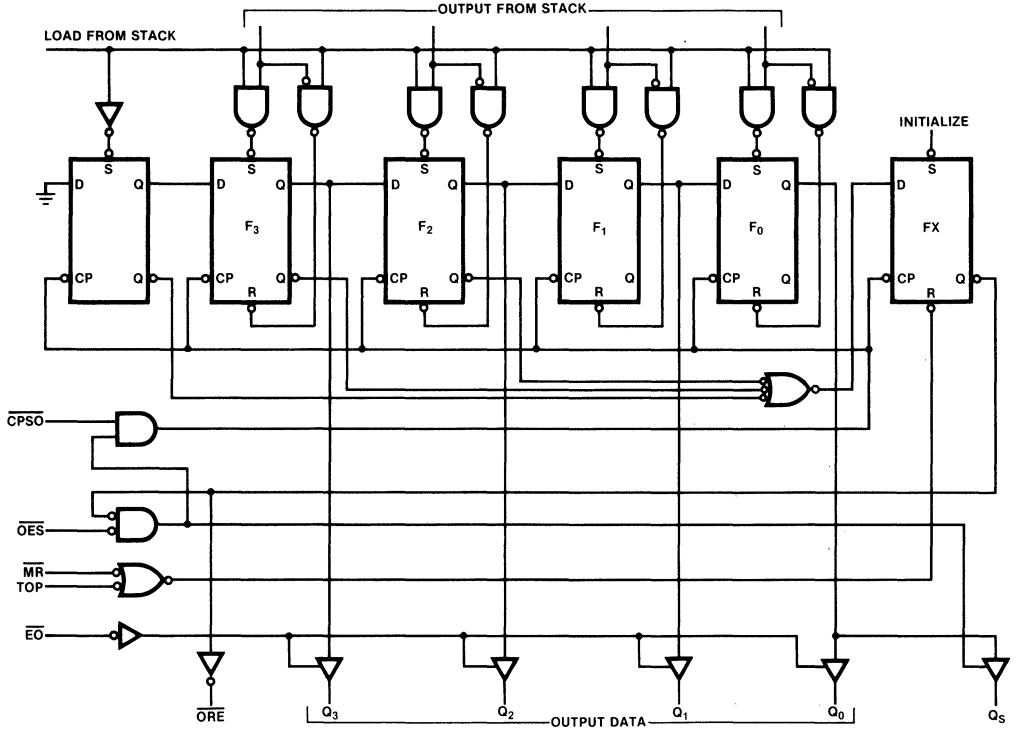


Figure 4 Output Register Conceptual Logic Diagram



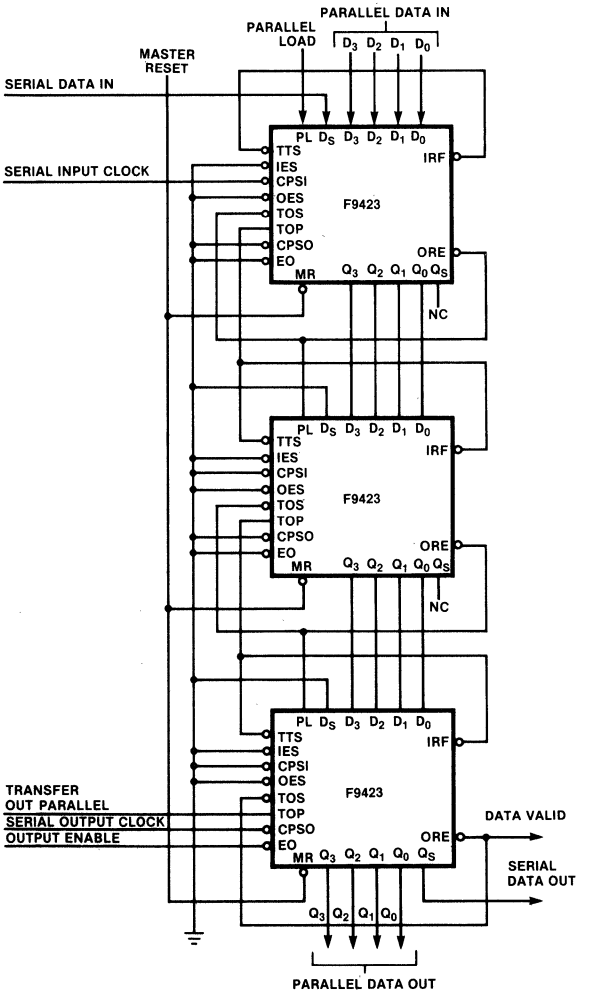
Parallel Extraction

When the FIFO is empty after a low pulse is applied to the $\overline{MR}$ input, the output register empty ($\overline{ORE}$) output is low. After data has been entered into the FIFO and has fallen through to the bottom stack location, it is transferred into the output register, if the transfer out parallel ($\overline{TOP}$) input is high. As a result of the data transfer, $\overline{ORE}$ goes high, indicating valid data on the data outputs (provided that the 3-state buffer is enabled). The TOP input can then be used to clock out the next word.

When TOP goes low, $\overline{ORE}$ also goes low, indicating that the output data has been extracted; however, the data itself remains on the output bus until a high level on $\overline{TOP}$ permits the transfer of the next word (if available) into the output register. During parallel data extraction, the serial output clock ($\overline{CPSO}$) line should be low. The transfer out serial ($\overline{TOS}$) line should be grounded for single-slice operation or connected to the appropriate $\overline{ORE}$ line for expanded operation (refer to the "Expansion" section).

The TOP signal is not edge-triggered. Therefore, if TOP goes high before data is available from the stack but data becomes available before TOP again goes low, that data is transferred into the output register. However, internal control circuitry prevents the same data from being transferred twice. If TOP goes high and returns to low before data is available from the stack, $\overline{ORE}$ remains low, indicating that there is no valid data at the outputs.

Figure 5 190-Word × 4-Bit Vertical Expansion Scheme



Serial Extraction

When the FIFO is empty after a low is applied to the $\overline{MR}$ input, the $\overline{ORE}$ output is low. After data has been entered into the FIFO and has fallen through to the bottom stack location, it is transferred into the output register, if the $\overline{TOS}$ input is low and TOP is high. As a result of the data transfer, $\overline{ORE}$ goes high, indicating that valid data is in the register.

The 3-state serial data output (Q_S) is automatically enabled and puts the first data bit on the output bus. Data is serially shifted out on the high-to-low transition of $\overline{CPSO}$. To prevent false shifting, $\overline{CPSO}$ should be low when the new word is being loaded into the output register. The fourth transition empties the shift register, forces $\overline{ORE}$ low, and disables the serial output, Q_S. For serial operation, the $\overline{ORE}$ output may be tied to the $\overline{TOS}$ input, requesting a new word from the stack as soon as the previous one has been shifted out.

Expansion

Vertical Expansion

The F9423 may be vertically expanded, without external components, to store more words. The interconnections necessary to form a 190-word by 4-bit FIFO are shown in figure 5. Using the same technique, any FIFO of 63n + 1 words by 4 bits can be configured, where n is the number of devices. Note that expansion does not sacrifice any of the F9423 flexibility for serial/parallel input and output.

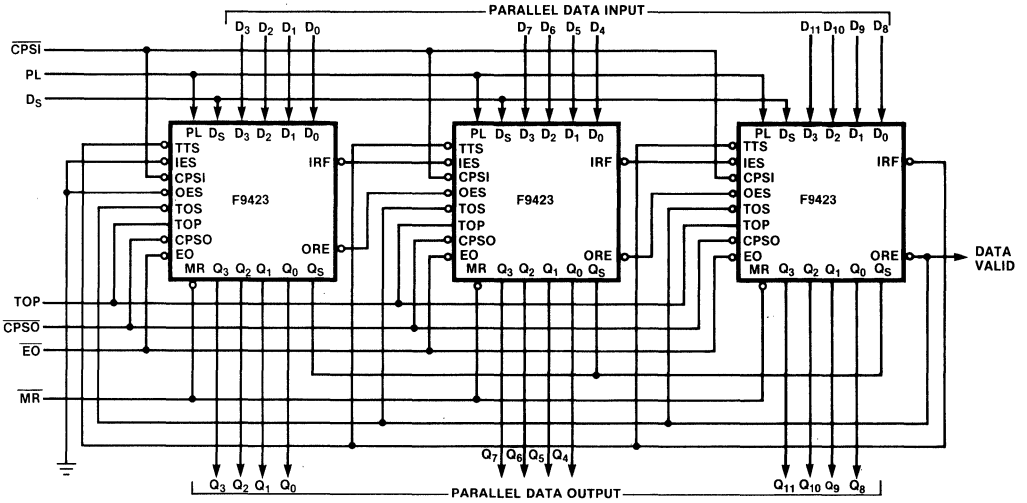
Horizontal Expansion

The F9423 can be horizontally expanded, without external logic, to store long words (in multiples of 4 bits). The interconnections necessary to form a 64-word by 12-bit FIFO are shown in figure 6. Using the same technique, any FIFO of 64 words by 4n bits can be constructed, where n is the number of devices.

The right-most (most significant) device is connected to the $\overline{TTS}$ inputs of all devices. Similarly, the $\overline{ORE}$ output of the most significant device is connected to the $\overline{TOS}$ inputs of all devices. As in the vertical expansion scheme, horizontal expansion does not sacrifice any of the F9423 flexibility for serial/parallel input and output.

It should be noted that the horizontal expansion scheme shown in figure 6 exacts a penalty in speed.

Figure 6 64-Word × 12-Bit Horizontal Expansion Scheme



Horizontal and Vertical Expansion

The F9423 can be expanded in both the horizontal and vertical directions without any external components and without sacrificing any of its FIFO flexibility for serial/parallel input and output. The interconnections necessary to form a 127-word by 16-bit FIFO are shown in figure 7. Using the same technique, any FIFO of 63m + 1 words by 4n bits can be configured, where m is the number of devices in a column and n is the number of devices in a row. Figures 8 and 9 illustrate the timing diagrams for serial data entry and extraction for the FIFO shown in figure 7. Figure 10 illustrates the final positions of bits in an expanded F9423 FIFO resulting from a 2032-bit serial bit train.

Interlocking Circuitry

Most conventional FIFO designs provide status signals analogous to $\overline{IRF}$ and $\overline{ORE}$. However, when these devices are operated in arrays, variations in unit-to-unit operating speed require external gating to ensure that all devices have completed an operation. The F9423 incorporates simple but effective "master/slave" interlocking circuitry to eliminate the need for external gating.

In the F9423 array of figure 7, devices 1 and 5 are the row masters; the other devices are slaves to the master in their rows. No slave in a given row initializes its input register until it has received a low on its $\overline{IES}$ input from a row master or a slave of higher priority.

Similarly, the $\overline{ORE}$ outputs of slaves do not go high until their inputs have gone high. This interlocking scheme ensures that new input data may be accepted by the array when the $\overline{IRF}$ output of the final slave in that row goes high and that output data for the array may be extracted when the $\overline{ORE}$ output of the final slave in the output row goes high.

The row master is established by connecting its $\overline{IES}$ input to ground, while a slave receives its $\overline{IES}$ input from the $\overline{IRF}$ output of the next-higher priority device. When an array of F9423 FIFOs is initialized with a low on the $\overline{MR}$ inputs of all devices, the $\overline{IRF}$ outputs of all devices are high. Thus, only the row master receives a low on the $\overline{IES}$ input during initialization.

Figure 7 127-Word × 16-Bit FIFO Array

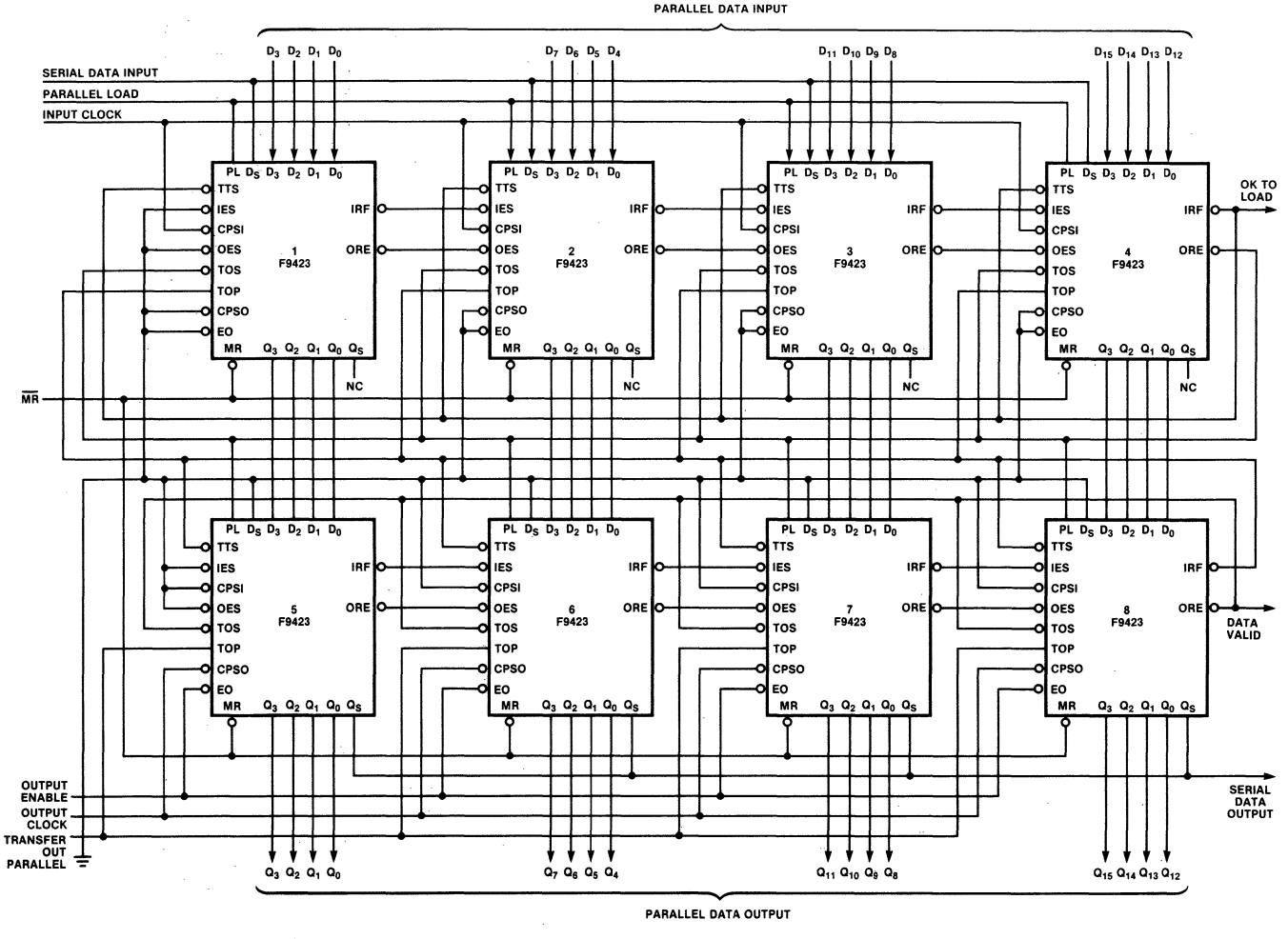


Figure 8 Serial Data Entry for 127 × 16 Array

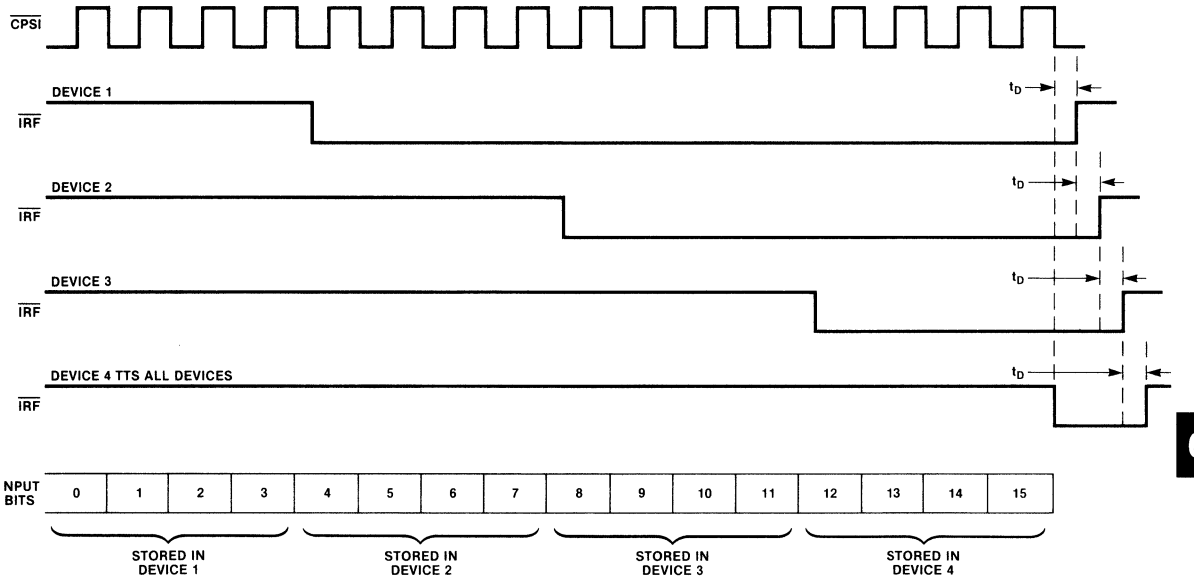


Figure 9 Serial Data Extraction for 127 × 16 Array

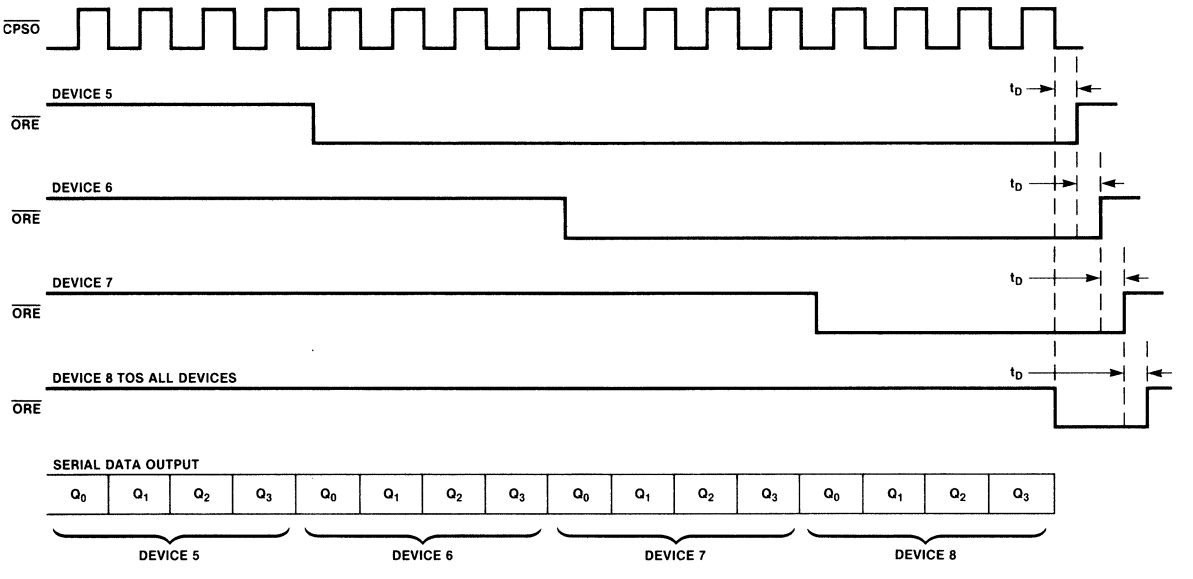


Figure 10 Final Position of 2032-Bit Serial Input

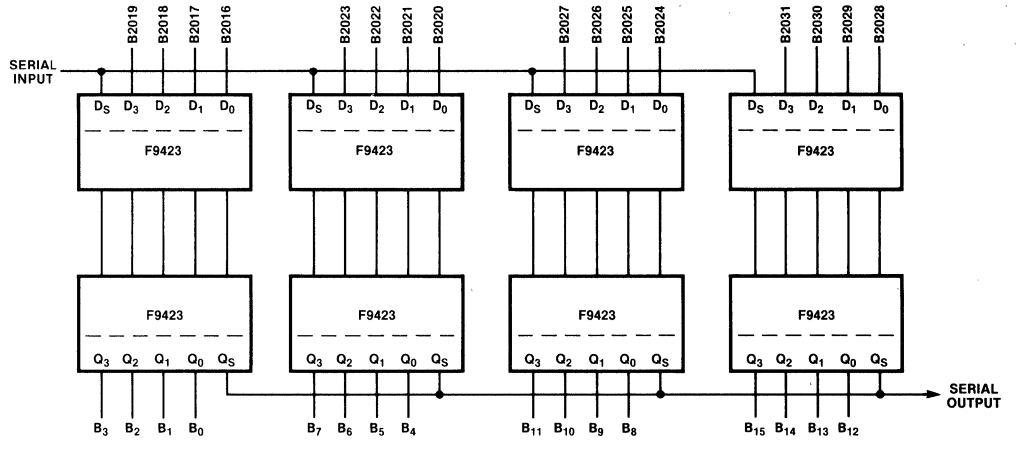


Figure 11 Interlocking Circuitry Conceptual Logic Diagram

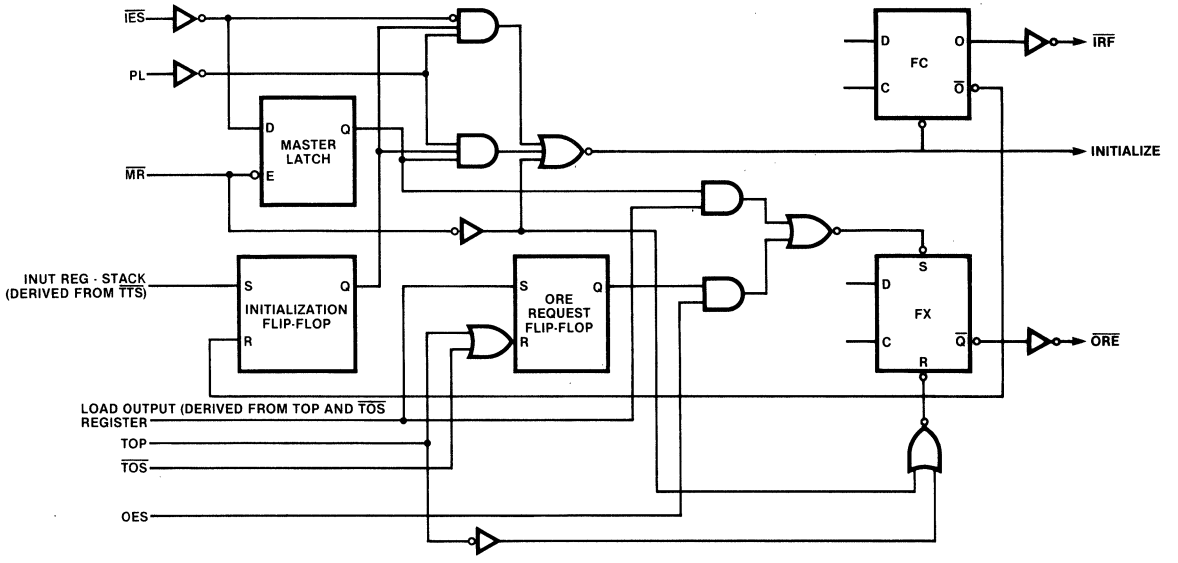


Figure 11 is a conceptual logic diagram of the internal circuitry that determines master/slave operation. When $\overline{MR}$ and $\overline{IES}$ are low, the master latch is set. When $\overline{TTS}$ goes low, the initialization flip-flop is set. If the master latch is high, the input register is immediately initialized and the initialization flip-flop reset. If the master latch is reset, the input register is not initialized until $\overline{IES}$ goes low. In array operaton, activating $\overline{TTS}$ initiates a ripple input register initialization from the row master to the last slave.

A similar operation takes place for the output register. Either a $\overline{TOS}$ or TOP input initiates a load-from-stack operation and sets the $\overline{ORE}$ request flip-flop. If the master latch is set, the last output register flip-flop is set and the $\overline{ORE}$ line goes high. If the master latch is reset, the $\overline{ORE}$ output is low until a serial output enable ($\overline{OES}$) input is received.

Timing Characteristics

Table 2 describes, and figures 12 through 19 illustrate, the F9423 timing characteristics.

Table 2 Timing Characteristics

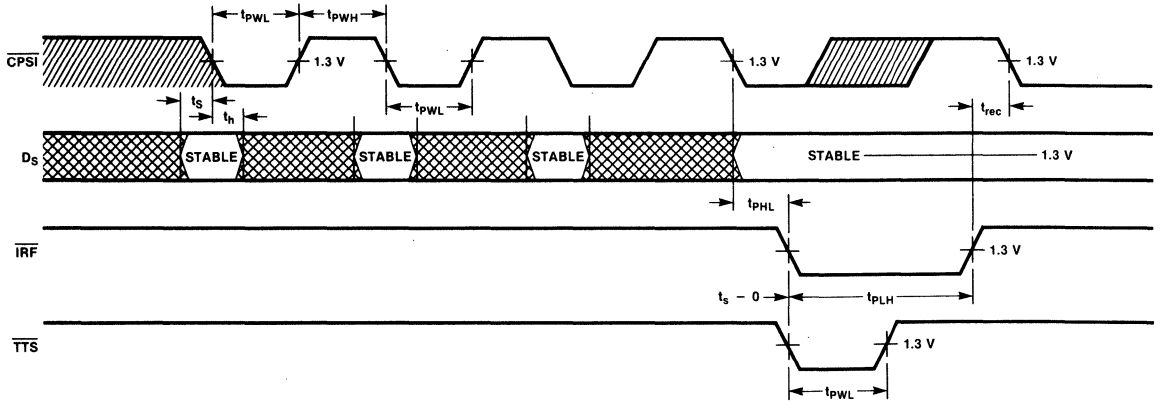
Symbol	Characteristic[1]	Limits			Units	Comments
		Min	Typ	Max		
t_{PHL}	Propagation Delay, Negative-Going CP to $\overline{IRF}$ Output		30	40	ns	Stack not full, PL low (see figures 12 and 13).
t_{PLH}	Propagation Delay, Negative-Going $\overline{TTS}$ to $\overline{IRF}$		68	90	ns	
t_{PLH}	Propagation Delay, Negative-Going $\overline{CPSO}$ to Q_S Output		46	55	ns	$\overline{OES}$ low, TOP high (see figures 14 and 15).
t_{PHL}			30	40	ns	
t_{PLH}	Propagation Delay, Positive-Going TOP to Q_0-Q_3 Outputs		80	95	ns	$\overline{EO}$, $\overline{CPSO}$ low (see figure 16).
t_{PHL}			68	80	ns	
t_{PHL}	Propagation Delay, Negative-Going $\overline{CPSO}$ to $\overline{ORE}$		29	50	ns	$\overline{OES}$ low, TOP high (see figures 14 and 15).
t_{PHL}	Propagation Delay, Negative-Going TOP to $\overline{ORE}$		39	60	ns	Parallel output, $\overline{EO}$, $\overline{CPSO}$ low (see figure 16).
t_{PLH}	Propagation Delay, Positive-Going TOP to $\overline{ORE}$		79	95	ns	
t_{DFT}	Fall-Through Time		3.6	4.3	μs	$\overline{TTS}$ connected to $\overline{IRF}$; $\overline{TOS}$ connected to $\overline{ORE}$; $\overline{IES}$, $\overline{OES}$, $\overline{EO}$, $\overline{CPSO}$ low, TOP high (see figure 17).
t_{PLH}	Propagation Delay, Negative-Going $\overline{TOS}$ to Positive-Going $\overline{ORE}$		72	85	ns	Data in stack, TOP high (see figures 14 and 15).

Table 2 Timing Characteristics (Continued)

Symbol	Characteristics[1]	Limits			Units	Comments
		Min	Typ	Max		
t_{PHL}	Propagation Delay, Positive-Going PL to Negative-Going $\overline{IRF}$		39	50	ns	Stack not full (see figures 18 and 19).
t_{PLH}	Propagation Delay, Negative-Going PL to Positive-Going $\overline{IRF}$		41	55	ns	
t_{PLH}	Propagation Delay, Positive-Going $\overline{OES}$ to $\overline{ORE}$		38	45	ns	
t_{PLH}	Propagation Delay, Positive-Going $\overline{IES}$ to Positive-Going $\overline{IRF}$		32	45	ns	See figure 19.
t_{PZL} t_{PZH}	Propagation Delay, $\overline{OE}$ to Q_0-Q_3		14	18	ns	Propagation delay out of the high-impedance state.
t_{PHZ} t_{PLZ}	Propagation delay, $\overline{OE}$ to Q_0-Q_3		16	20	ns	Propagation delay into the high-impedance state.
t_{PZL} t_{PZH}	Propagation Delay, Negative-Going $\overline{OES}$ to Q_S		14	20	ns	Propagation delay out of the high-impedance state.
t_{PLZ} t_{PHZ}	Propagation Delay, Negative-Going $\overline{OES}$ to Q_S		16	22	ns	Propagation delay into the high-impedance state.
t_{AP}	Parallel Appearance Time, $\overline{ORE}$ to Q_0-Q_3		4	6	ns	Time elapsed between $\overline{ORE}$ going high and valid data appearing at output. Negative number indicates data available before $\overline{ORE}$ goes high.
t_{AS}	Serial Appearance Time, $\overline{ORE}$ to Q_S		5	18	ns	

[1] V_{CC} = 5.0 V ±5%; C_L = 15 pF; T_A = 0°C to +75°C

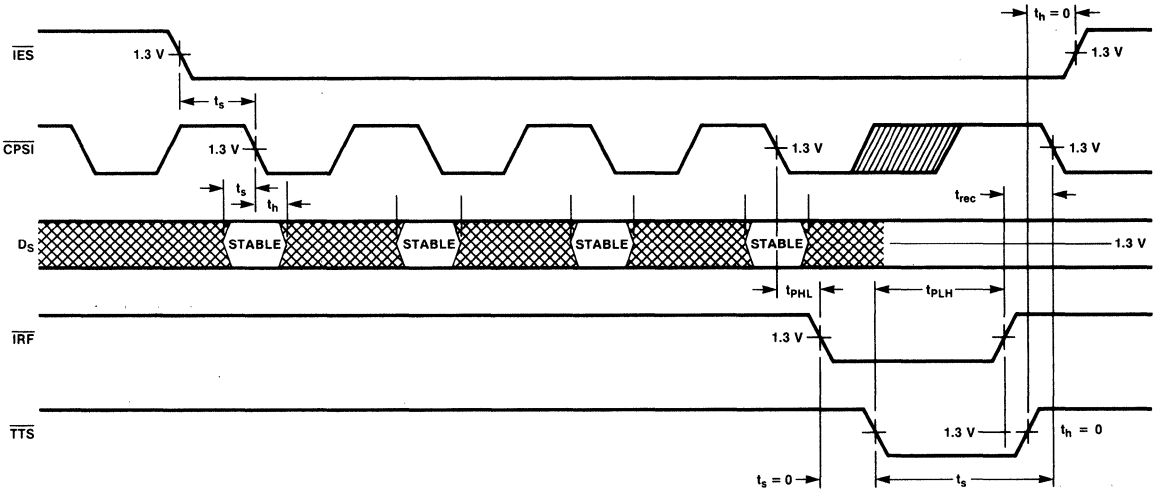
Figure 12 Serial Input, Unexpanded or Master Operation



Conditions: Stack Not Full; $\overline{IES}$, PL Low

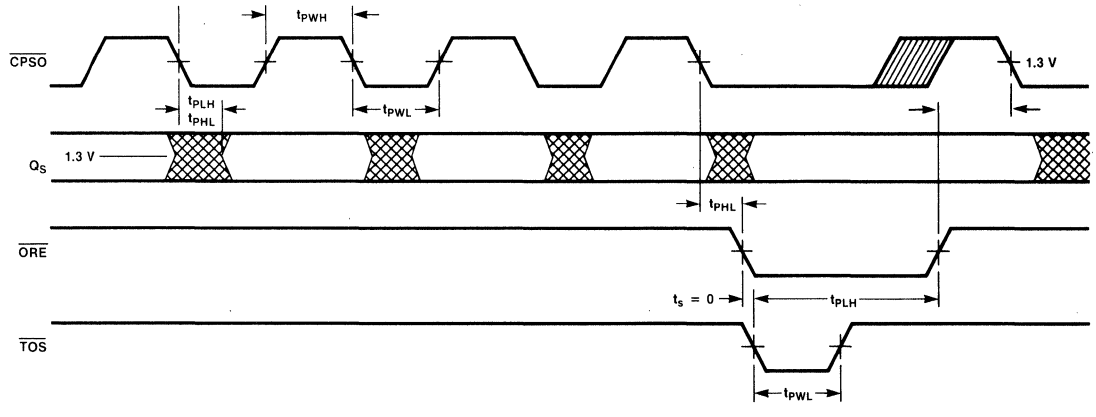
6

Figure 13 Serial Input, Expanded Slave Operation



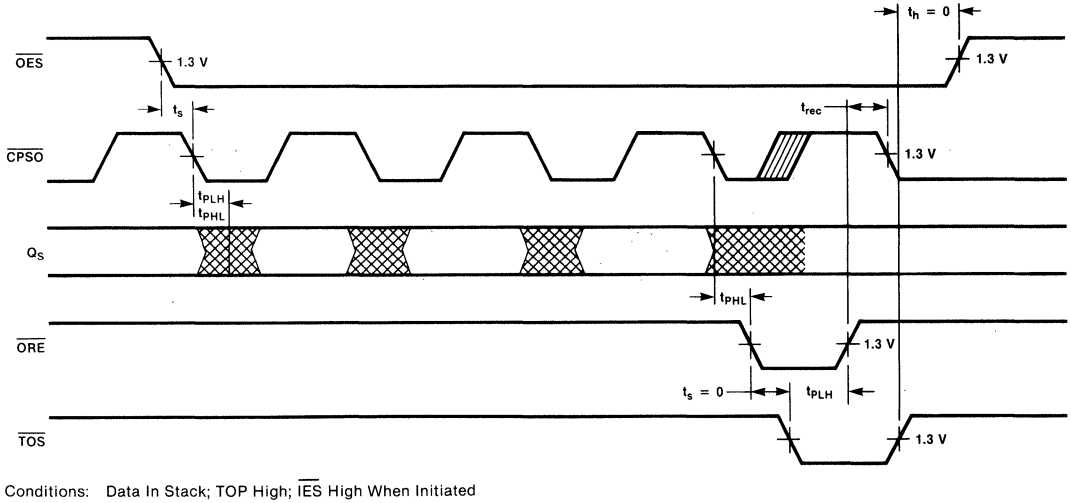
Conditions: Stack Not Full; $\overline{IES}$ High When Initiated; PL Low

Figure 14 Serial Output, Unexpanded or Master Operation



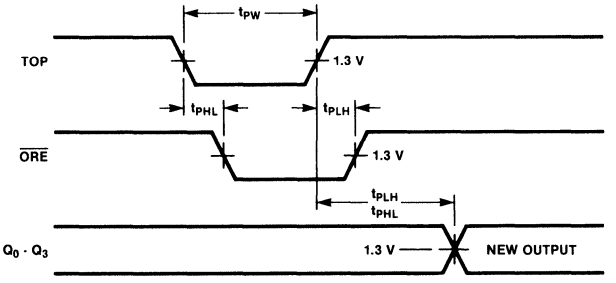
Conditions: Data in Stack; TOP High, $\overline{IES}$ Low When Initiated; $\overline{OES}$ Low

Figure 15 Serial Output, Slave Operation



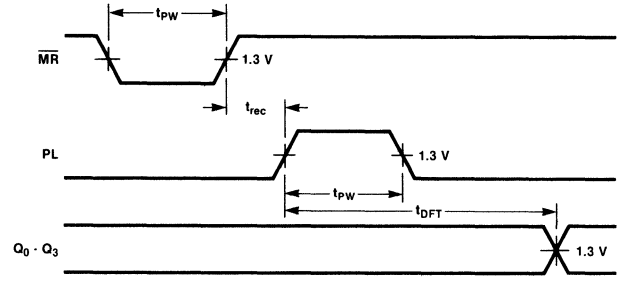
Conditions: Data In Stack; TOP High; $\overline{IES}$ High When Initiated

Figure 16 Parallel Output, 4-Bit Word or Master in Parallel in Expansion



Conditions: $\overline{IES}$ Low When Initiated; $\overline{EO}$, $\overline{CPSO}$ Low; Data Available In Stack

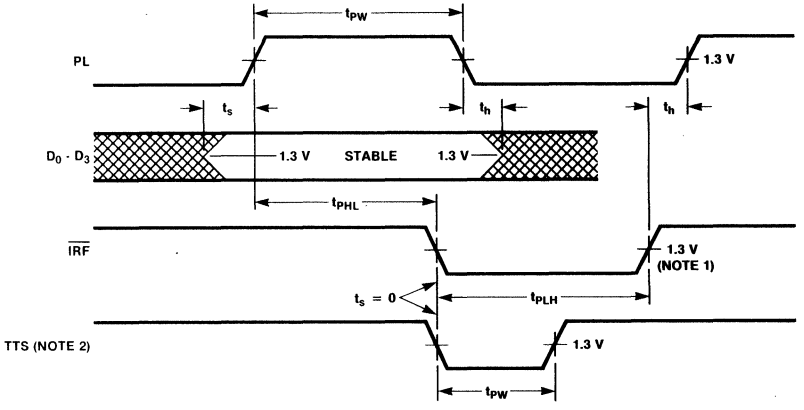
Figure 17 Fall-Through Time



Conditions: $\overline{TTS}$ Connected to $\overline{IRF}$; $\overline{TOS}$ Connected to $\overline{ORE}$; $\overline{IES}$, $\overline{OES}$, $\overline{CPSO}$ Low; TOP High

6

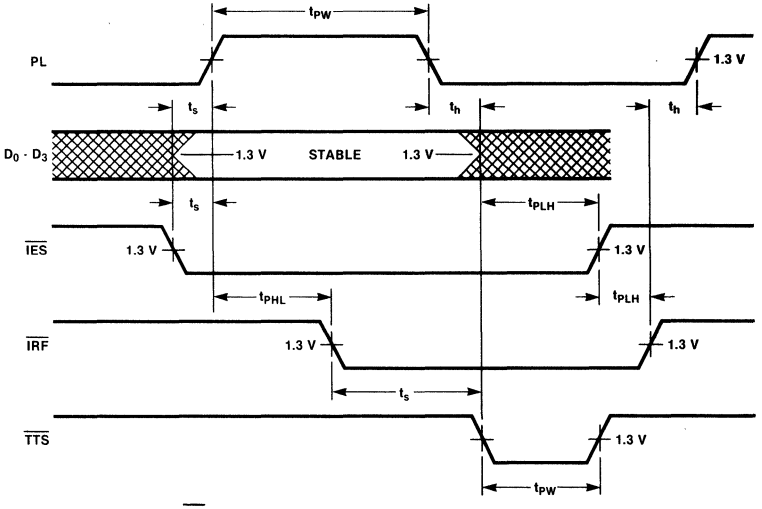
Figure 18 Parallel Load Mode, 4-Bit Word (Unexpanded) or Master in Parallel Expansion



NOTES:
[1] If stack is full, $\overline{IRF}$ stays low.
[2] $\overline{TTS}$ normally connected to $\overline{IRF}$.

Conditions: Stack Not Full; $\overline{IES}$ Low When Initialized

Figure 19 Parallel Load, Slave Mode



Conditions: Stack Not Full; Device Initialized With $\overline{IES}$ High;
Initialization Requires That a Master Reset Occur After
Power Is Applied

Timing Set-Up Requirements

Table 3 describes the F9423 timing set-up requirements.

Table 3 Timing Set-Up Requirements

Symbol	Characteristics[1]	Limits			Units	Comments
		Min	Typ	Max		
t_{PWH}	$\overline{CPSI}$ Pulse Width (High)	18	15		ns	Stack not full; PL low (see figures 12 and 13).
t_{PWL}	$\overline{CPSI}$ Pulse Width (Low)	22	15		ns	
t_{PWH}	PL Pulse Width (High)	11	10		ns	Stack not full (see figures 18 and 19).
t_{PWL}	$\overline{TTS}$ Pulse Width (Low) Serial or Parallel Mode	40	26		ns	Stack not full; (see figures 12, 13, 18, 19).
t_{PWL}	$\overline{MR}$ Pulse Width (Low)	35	22		ns	See figure 17.
t_{PWH}	TOP Pulse Width (High)	52	35		ns	$\overline{CPSO}$ low; data available in stack (see figure 16).
t_{PWL}	TOP Pulse Width (Low)	32	24		ns	
t_{PWH}	$\overline{CPSO}$ Pulse Width (High)	18	11		ns	TOP high; data in stack.
t_{PWL}	$\overline{CPSO}$ Pulse Width (Low)	25	17		ns	See figures 14 and 15.
t_s	Set-Up Time, D_S to Negative $\overline{CPSI}$	10	6		ns	PL low (see figures 12 and 13).
t_h	Hold Time, D_S to $\overline{CPSI}$	6	4		ns	PL low (see figures 12 and 13).
t_s	Set-Up Time, $\overline{TTS}$ to $\overline{IRF}$ Serial or Parallel Mode	1	−17		ns	See figures 12, 13, 18, 19.
t_s	Set-Up Time, Negative-Going $\overline{ORE}$ to Negative-Going $\overline{TOS}$	0	−26		ns	TOP high (see figures 14 and 15).
t_{rec}	Recovery Time, $\overline{MR}$ to Any Input	30	24		ns	See figure 17.
t_s	Set-Up Time, Negative-Going $\overline{IES}$ to $\overline{CPSI}$	18	15		ns	See figure 13.
t_s	Set-Up Time, Negative-Going $\overline{TTS}$ to $\overline{CPSI}$	110	83		ns	See figure 13.
t_s	Set-Up Time, Parallel Inputs to PL	0	−12		ns	Length of time parallel inputs must be applied prior to rising edge of PL.
t_h	Hold Time, Parallel Inputs to PL	20	10		ns	Length of time parallel inputs remain applied after falling edge of PL.

V_{CC} = 5.0 V; C_L = 15 pF; T_A = 25 °C

DC Characteristics

Table 4 describes the F9423 dc characteristics.

Table 4 DC Characteristics

Symbol	Characteristic[1]		Limits[2] Min	Typ	Max	Units	Test Conditions[3]
V_{IH}	Input High Voltage		2.0			V	Guaranteed input high voltage
V_{IL}	Input Low Voltage				0.8	V	Guaranteed input low voltage
V_{CD}	Input Clamp Diode Voltage			−0.9	−1.5	V	V_{CC} = Min; I_{IN} = −18 mA
V_{OH}	Output High Voltage, $\overline{ORE}$, $\overline{IRF}$		2.4	3.4		V	V_{CC} = Min; I_{OH} = −400 μA
V_{OH}	Output High Voltage, Q_0-Q_3, Q_S		2.4	3.1		V	V_{CC} = Min; I_{OH} = −5.7 mA
V_{OL}	Output Low Voltage, $\overline{ORE}$, $\overline{IRF}$			0.35	0.5	V	V_{CC} = Min; I_{OL} = 8.0 mA
V_{OL}	Output Low Voltage, Q_0-Q_3, Q_S			0.35	0.5	V	V_{CC} = Min; I_{OL} = 16 mA
I_{OZH}	Output Off High Current, Q_0-Q_3, Q_S				100	μA	V_{CC} = Max; V_{OUT} = 2.4 V; V_E = 2.0 V
I_{OZL}	Output Off Low Current, Q_0-Q_3, Q_S				−100	μA	V_{CC} = Max; V_{OUT} = 0.5 V; V_E = 2.0 V
I_{IH}	Input High Current			1.0	40	μA	V_{CC} = Max; V_{IN} = 2.7 V
					1.0	mA	V_{CC} = Max; V_{IN} = 5.5 V
I_{IL}	Input Low Current, Except $\overline{OES}$, $\overline{MR}$				−0.4	mA	V_{CC} = Max; V_{IN} = 0.4 V
	Input Low Current, $\overline{OES}$, $\overline{MR}$				−0.8	mA	
I_{OS}[4]	Output Short Circuit Current	$\overline{ORE}$, $\overline{IRF}$	−15		−100	mA	V_{CC} = Max; V_{OUT} = 0 V
		Q_0-Q_3, Q_S	−30		−130	mA	
I_{CC}	Supply Current			140	180	mA	V_{CC} = Max; inputs open

Notes

1. Typical limits are at V_{CC} = 5.0 V ±5%, T_A = 0°C to +75°C, and Max loading. The temperature ranges are guaranteed with transverse air flow exceeding 400 linear feet per minute. Typical thermal resistance values of the package at maximum temperature are:

   θ_{JA} (junction-to-ambient) at 400 fpm air flow = 50°C/W, ceramic DIP; 65° C/W, plastic DIP.
   θ_{JA} (junction-to-ambient) in still air = 90°C/W, ceramic DIP; 110°C/W, plastic DIP.
   θ_{JC} (junction-to-case) = 25°C/W, ceramic DIP; 25°C/W, plastic DIP.

2. The specified limits represent the worst-case values for the characteristic. Since these values normally occur at the temperature and supply voltage extremes, additional noise immunity and guard banding can be achieved by decreasing the allowable system operating ranges.

3. Conditions for testing not shown in the table are chosen to guarantee operation under worst-case conditions.

4. Duration of short circuit should not exceed 1 second; not more than one output should be shorted at a time.

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this document, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Storage Temperature	−65 °C, +150 °C
Temperature (Ambient) Under Bias	−55 °C, +125 °C
V_{CC} Pin Potential to Ground Pin	−0.5 V, +6.0 V
*Input Voltage (DC)	−0.5V, +5.5 V
*Input Current (DC)	−12 mA, +5.0 mA
**Voltage Applied to Outputs (Output High)	−0.5V, +5.5V
Output Current (Output Low)	+20 mA

*Either input voltage or input current limit is sufficient to protect the input.
**Output current limit required.

Guaranteed Operating Ranges

Part Number	Supply Voltage (V_{CC})			Ambient Temperature (T_A)
	Min	Typ	Max	
F9423XC	4.75 V	5.0 V	5.25 V	0 °C to +75 °C

X = package type (D for ceramic DIP, P for plastic DIP)

FAIRCHILD
A Schlumberger Company

F9443
Floating-Point Processor

Advance Product Information | Microprocessor Product

Description

The F9443 Floating-Point Processor is designed to provide enhancement to the numeric capabilities of 16-bit microprocessors by providing a set of floating-point instructions. It can interface with the F9445, F9450 or any other standard 16-bit microprocessor, and it uses the microprocessor memory to directly fetch the required operands. It has eight general-purpose registers on-chip and supports all the basic functions with on-chip microcode. Use of additional off-chip microcode read-only memories (ROMs) provides extended capabilities. Figure 1 is a block diagram of the F9443.

Circuit Description

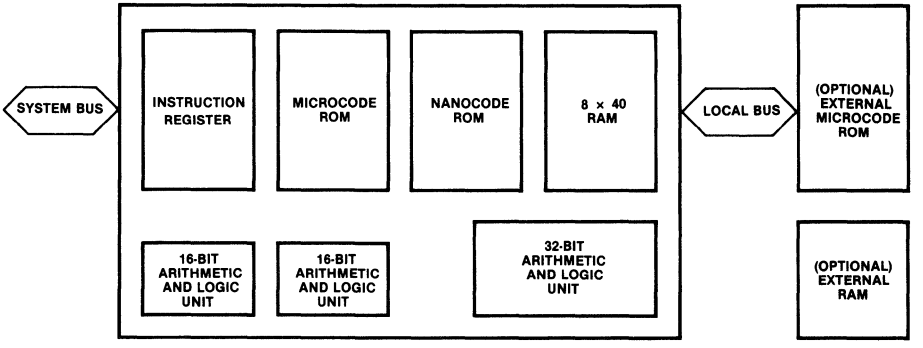
The F9443 includes special hardware to provide fast algorithms for the basic functions. This hardware includes full-carry look-ahead for add and subtract (ADD/SUB) functions, recoding logic for multiply and square root functions, and partial-remainder-prediction logic for divide functions. An advanced control scheme provides a 2-level microcode/nanocode control with off-chip microcode expansion. The off-chip microcode can be programmable ROM (PROM) or random-access memory (RAM), with easy expansion for fast implementation of user algorithms.

Operation of the F9443 can proceed in parallel with the host processor to maximize throughput. Multiple F9443s can be connected to the host processor for array processing or other high-speed applications.

- **Full IEEE SP D DX Floating-Point Standard Support (80 Bits)**
- **Fast Algorithms for Add, Subtract, Multiply, Square Root, and Divide Functions**
- **Support for Full Set of Trigonometric Exponential and Logarithmic Functions**
- **Expandable Instruction Set That Can Include Macro Operations (e.g., Vector rotate, Fourier Transform, Array and Matrix Applications)**
- **User-Alterable Microcode for User Functions**
- **Support of Integer Decimal and Logical Functions**
- **Standard 64-Pin Package**
- **I^3L® High Speed Bipolar Logic**
- **Low-Power Schottky-Compatible I/O**
- **Very Fast Execution Times**
- **Interface to Any 16-Bit Microprocessor**

6

Figure 1 9443 Block Diagram



® I^3L is a registered trademark of Fairchild Camera and Instrument Corp.

FAIRCHILD
A Schlumberger Company

F9444
Memory Management and Protection Unit

Advance Product Information

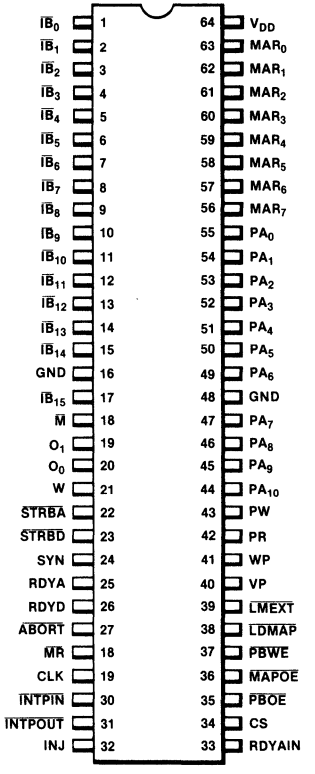
Microprocessor Product

Description

The Fairchild F9444 programmable Memory Management and Protection Unit (MMPU) is designed to support complex multi-user and large single-user environments. With four F93422 bipolar static random-access memories (RAMs) serving as map memory, the F9444 expands the physical address space of the F9445 16-bit microprocessor to 2M words by performing logical-to-physical address translation. That is, the six most significant bits (MSBs) of the logical address are translated into 11 physical address bits, leaving the 10 least significant bits (LSBs) of the logical address unchanged. The memory thus consists of 21 bits (10 LSB and 11 MSB), or 4 megabytes. System integrity is maintained by access protection bits associated with each page. Any violation causes non-maskable interrupt to the F9445 central processing unit. Page-written (PW) and page-referenced (PR) bits permit the implementation of demand-paging algorithms. Figure 1 is a functional diagram of the MMPU.

- **Standard Input/Output (I/O) Instruction Format**
- **Ability to Implement Demand-Paged Virtual Memory System**
- **Ability to Access Up to 2M Words of Memory**
- **2K Pages, With 1K Words for Each Page**
- **Memory Expansion Through Mapping and Demand Paging**
- **Controls for Memory Mapping**
- **Separate RAMs for Storing Maps**
- **Access and I/O Protection to Maintain System Integrity**
- **Special Status Bits for Read/Write Protection, Demand Paging, and I/O Protection**
- **Support for Two User and Two Data Channel Maps**
- **Low-Power Schottky-Compatible I/O**
- **Single +5 V Power Supply**
- **64 Pin Dual-in-Line Package (DIP)**
- **I^3L® Technology**

Connection Diagram



6

® I^3L is a registered trademark of Fairchild Camera and Instrument Corp.

Maps

The MMPU allows two user maps and two data channel maps to reside in map memory. Each data channel map contains 32 1K-word pages and each user map either 32 or 64 1K-word pages that can be relocated anywhere in memory. The two user maps and two data channel maps function independently. Only one user map can be enabled at a time, but both data channel maps are enabled at the same time. The supervisor determines whether the mapping of program address and data channel address are to be enabled at the same time. If either user mapping or data channel mapping is disabled, the physical address space for that function is equal to the logical address space and only the lowest 64K words can be accessed.

Signal Functions

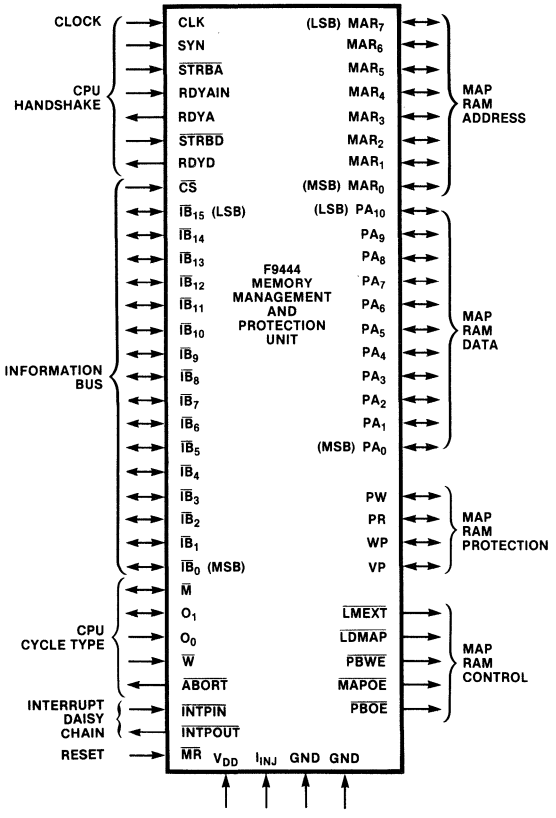
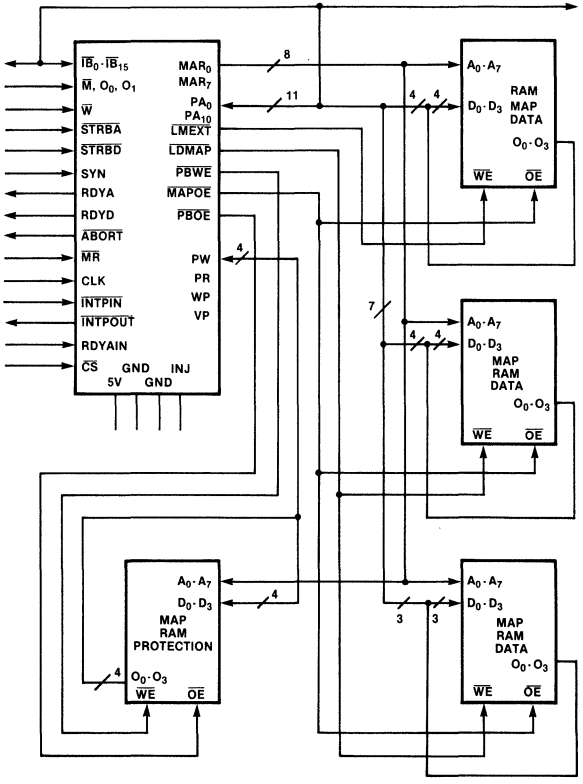


Figure 1 F9444 Functional Diagram



FAIRCHILD
A Schlumberger Company

F9445
16-Bit Bipolar Microprocessor

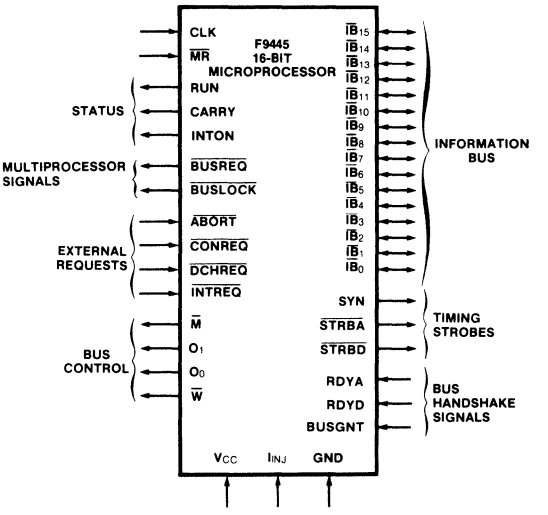
Microprocessor Products

Description

The F9445 is a 16-bit microprocessor implemented using Fairchild's Isoplanar Integrated Injection Logic ($I^3L^®$) technology. This bipolar technology and a sophisticated pipeline architecture combine to give the F9445 very fast execution times. The processor has eight program-accessible registers and the capability of directly addressing 128K bytes (64K words) of memory. Up to 4M bytes of physical memory may be accessed using the F9444 memory management unit. The F9445 can address 62 I/O devices, handle 16 levels of priority interrupt, and perform fast direct memory access. It has control lines to provide operator-console functions and has an on-chip self-test program. The F9445 CPU is supported with a comprehensive family of LSI support circuits to permit cost and performance effective usage in high-performance microcomputer systems. The support circuits include the F9446 Dynamic Memory Controller, F9447 I/O Controller, F9448 Programmable Multiport Interface, F9449 Multiple Data Channel Controller, F9444 Memory Management Unit and F9470 Console Controller. It is also supported with a library of software packages, including editors, debuggers, macro-assembler, relocating loader, real-time executive, interactive multi-user disk operating system and utilities, as well as high-level languages: FORTRAN, BASIC and PASCAL.

- **Advanced Parallel Architecture Leading to Very Fast Execution Times—250 ns Register to Register, 2.9 μs 16 × 16 Bit Multiply**
- **Directly Addresses up to 128K Bytes of Memory with 11 Addressing Modes**
- **Eight Program-Accessible Registers (AC0, AC1, AC2, AC3, SP, FP, PC, PSW)**
- **Versatile Instruction Set Including Memory Reference, ALU, I/O, Stack, Multiply/Divide, and Floating Point Assist (Scale/Normalize) Instructions with 8-Bit Byte, 16-Bit Word or 32-Bit Double-Word Data**
- **Multi-Processing Capabilities**
- **Flexible Operator-Control Functions and Self-Test**
- **Static Operation with Single Clock up to 24 MHz**
- **LS TTL Input/Output Structure with I^3L Internal Circuits**
- **40-Pin DIP Needing a Single +5 V Power Supply**
- **Full Military Temperature and Voltage Ranges**
- **Radiation-Tolerant Technology**
- **Comprehensive Family of Support Circuits**

Pin Functions



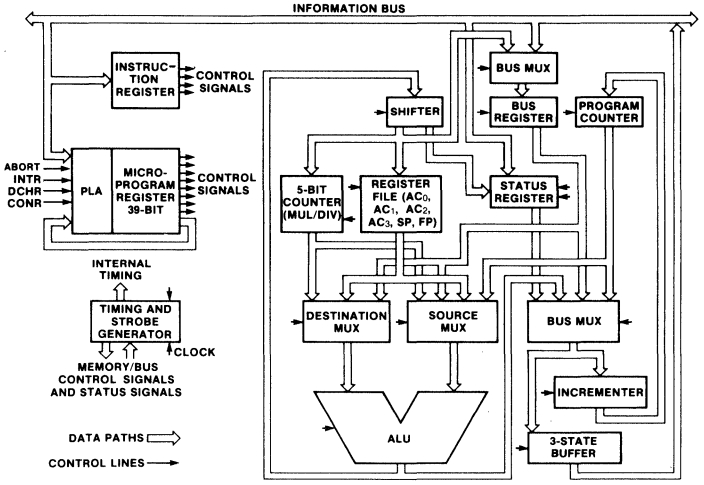
Absolute Maximum Ratings

Beyond these ratings useful life of the device may be impaired.

Storage Temperature	−65 to +150°C
Ambient Temperature Under Bias	−55 to +125°C
V_{CC} Pin Potential to Ground Pin	−0.5 to +6.0 V
Input Voltage (dc)	−0.5 to +5.5 V
Input Current (dc)	−20 to +5 mA
Output Voltage (Output HIGH)	−0.5 to +5.5 V
Output Current (dc) (Output LOW)	+20 mA
Injector Current (I_{INJ})	+450 mA
Injector Voltage (V_{INJ})	−0.5 to +1.5 V

6

Fig. 1 F9445 Functional Diagram



Architecture

The F9445 microprocessor comprises three main blocks: the data path, the control unit, and the timing generator.

Data Path

The data path is 16 bits wide and is responsible for all the processing of data and address in the system. In many cases, data and address may be processed simultaneously.

The data path includes the following blocks (see *Figure 1*):
Register File (AC0, AC1, AC2, AC3, SP, FP)
Program Counter (PC)
Program Status Word or Status Register (PSW containing: Carry, Overflow, 32KW, ETRP flags)
Interrupt-On Flip-Flop (INTON)
Destination Mux
Source Mux
16-Bit ALU
17-Bit Shifter
5-Bit Counter (for multicycle instructions)
Bus Register Mux
Bus Register
Bus Mux and Buffer
Incrementer

Control Unit

The operations of the data path components are governed by the pipelined, microprogrammed control unit. This unit comprises three main elements (see *Figure 1*): the PLA (control store) to contain the microprogram, the pipeline register (microprogram register) to latch the microinstruction executed in the current cycle, and the instruction register to supply additional control bits during certain instructions. In addition, the control unit has a machine instruction pre-fetch mechanism which overlaps the fetching of the next instruction from memory during execution of short-cycle instructions, such as arithmetic-and-logic (ALU) instructions. This pre-fetch capability and the microprogram pipeline give the F9445 very fast and efficient instruction execution.

Timing Generator

The timing generator produces the system timings for the F9445 internal registers, memory, I/O, and console.

The clock is divided on-chip using a 3-bit twisted ring counter. The divide ratio is 6:1 or 4:1, depending on whether a short or long cycle is required. The long cycle can be extended indefinitely by lowering the inputs BUSGNT, RDYA, or RDYD. These signals hold the processor in state S1 (using BUSGNT or RDYA) or S3 (using RDYD) until the inputs are raised.

The twisted ring counter is also used to generate all the strobes by a combinational decode of its outputs and certain bits of the microprogram register.

Signal Descriptions

All F9445 inputs and outputs are TTL.

Information Bus
$\overline{IB}_0$ through $\overline{IB}_{15}$, Pins 11 through 26 — 16-bit Bus — Active LOW bidirectional; $\overline{IB}_0$ is most significant bit; address valid

with $\overline{STRBA}$ strobe; data valid with $\overline{STRBD}$ strobe; 3-state during data-channel and non-bus cycles.

Timing and Status

SYN, Pin 7 — Synchronize Output — Active every cycle; may be used for external synchronization of memory and I/O control.

$\overline{STRBD}$, Pin 6 — Data Strobe — Active LOW output; active only during memory, I/O, console, or data-channel cycles; used as strobe for data.

$\overline{STRBA}$, Pin 5 — Strobe Memory Address Register — Active LOW output; active only during normal memory cycles; not active during write portion of read-modify-write cycles (DSZ, ISZ, STB instructions and auto-increment/decrement addressing modes); used as strobe for external address register; active on I/O cycles when I/O instruction is output onto bus.

$\overline{M}$, Pin 36 — Memory or I/O Function — Active LOW output.
O_1, Pin 35 — Memory or I/O Function — Active HIGH output.
O_0, Pin 34 — Memory or I/O Function — Active HIGH output; these pins indicate the type of bus transfer as shown in the following table.

	$\overline{M}$	O_1	O_0	Function
Memory	0	0	0	Instruction Fetch
	0	0	1	Operand
	0	1	0	Indirect Address
	0	1	1	Address Save on interrupt, abort, and trap
I/O	1	0	0	Input or Output
	1	0	1	Data Channel Acknowledge
	1	1	0	Read Console Code
	1	1	1	Console Data

If a skip is taken on an arithmetic-and-logic (ALU) instruction, the next instruction is fetched but not executed. In such fetches, the $\overline{M}$ and O lines will indicate the following states.

$\overline{M}$	O_1	O_0	State Indicated
0	0	0	S0 through S4
0	0	1	S5

During machine cycles that do not use the bus, the $\overline{M}$ and O lines will be "111". $\overline{BUSREQ}$ and the bus strobes are inactive in these cycles.

$\overline{W}$, Pin 1 — Write Output — Indicates direction of data flow; HIGH indicates a read or input operation; LOW indicates a write or output operation; 3-state during data-channel cycles and short cycles ($\overline{BUSREQ}$ is HIGH).

RDYD, Pin 8 — Data Ready — Active HIGH input; used to synchronize external devices with the F9445 during data transfer; a LOW level halts the processor.

RDYA, Pin 4 — Address Ready — Active HIGH input; maintains address on bus when LOW.

RUN, Pin 37 — Run Status — Active HIGH output; LOW when in halt state.

CARRY, Pin 39 — Carry Status — Active HIGH output; copy of carry bit.

INTON, Pin 27 — Interrupt-On Status — Active HIGH output; copy of Interrupt-On flag; HIGH when interrupts enabled.

CLK, Pin 40 — Clock Input — Single-phase clock; positive-edge triggered.

Arbitration

$\overline{BUSREQ}$, Pin 38 — Bus Request — Active LOW output; indicates that a bus cycle is required; useful in multi-microprocessor system.

$\overline{BUSLOCK}$, Pin 2 — Bus Lock — Active LOW open collector output; set during read portion of read-modify-write cycles (on DSZ, ISZ, STB, and auto-increment/decrement), reset during write portion of those cycles; used in multi-microprocessor system.

BUSGNT, Pin 3 — Bus Grant — Active HIGH input; used for multi-microprocessor operation; a LOW level inhibits address output and halts the processor.

Service Request

The order of priority of requests and interrupts, from highest to lowest, is as follows: $\overline{MR}$, $\overline{ABORT}$, $\overline{DCHREQ}$, Stack Overflow Interrupt, $\overline{INTREQ}$, and $\overline{CONREQ}$.

$\overline{MR}$, Pin 33 — Master Reset — Active LOW input; a LOW level causes the processor to enter a wait state after completing the next full cycle; if that cycle is a write, it is inhibited (changed to read); sets the F9445 to 32K mode with trap enabled.

$\overline{DCHREQ}$, Pin 29 — Data Channel Request — Active LOW input; initiates data-channel cycles while LOW after current instruction. Must occur before TDRH (c).

$\overline{CONREQ}$, Pin 28 — Console Request — Active LOW input; initiates a console operation after current instruction.

$\overline{\text{INTREQ}}$, Pin 30 — Interrupt Request — Active LOW input; initiates entry to interrupt procedure, if interrupts are enabled, after the current instruction.

$\overline{\text{ABORT}}$, Pin 32 — Abort — Active LOW input; initiates abort sequence in the current microcycle.

Power

V_{CC}, Pin 31 — Power Supply — Requires +5 V.

GND, Pin 9 — Ground.

I_{INJ}, Pin 10 — Injection Current Input — Operates in 200-400 mA range at approximately 1 V; requires > 350 mA for maximum speed.

Register Set

The F9445 has eight user-accessible registers (see Figure 2), including seven 16-bit registers and a program status word (PSW) containing the following four flags: carry (bit 0), 32KW (bit 1), trap enable (bit 2), and overflow (bit 15). The carry flag (C) indicates the state of the carry bit during arithmetic and logic operations. The 32KW flag indicates whether the processor is operating in the 32K-word ("1") or 64K-word ("0") mode. The trap enable/disable flag (ETRP) indicates whether the trap instruction is enabled ("1") or disabled ("0"). The overflow flag (V) indicates twos-complement overflow in arithmetic operations.

In addition, there is an interrupt-on (INTON) flag. The CPU responds to interrupt requests from external I/O devices when the flag is set ("1"). When it is clear ("0"), all interrupt requests are ignored by the CPU. The state of the flag can be altered by the Interrupt-Enable or Interrupt-Disable instruction.

The seven 16-bit registers comprise a program counter (PC) that sequences the execution of instructions, four general-purpose accumulators (AC0 through AC3), the stack pointer (SP) and the frame pointer (FP). The program counter sequences the execution of instructions. It holds the address of the next instruction to be executed and is automatically incremented to fetch instructions from consecutive memory locations. A Skip, Jump, Jump-to-Subroutine, or Trap instruction, an interrupt generated by an I/O device or an Abort can alter the sequential execution of instructions.

The four accumulators serve as source and destination registers for 16-bit arguments in arithmetic-and-logic instructions which process the contents of the source accumulators and a base value for the carry flag and store the 16-bit result in the destination accumulator. The associated carry and overflow flags are set or cleared depending on

Fig. 2 F9445 Register Model

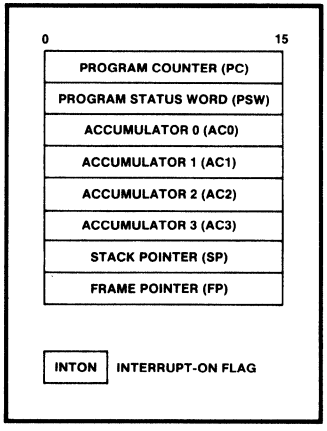
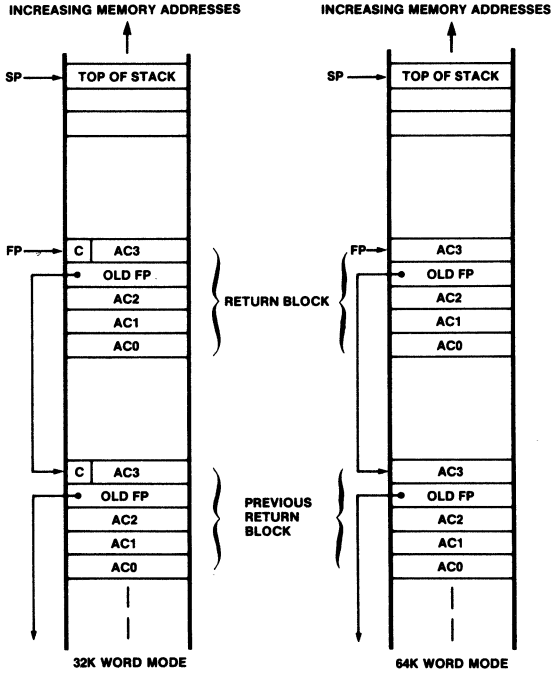


Fig. 3 Data Organization in a Stack (LIFO)



the result of the ALU operation as the base value of carry. Accumulators AC2 and AC3 also serve as index registers during memory addressing operations. In addition, AC3 functions as a subroutine linkage register, and the pair AC0 and AC1 are used as a 32-bit register in the multiply/divide and the normalize and parametric double-shift instructions.

The other two 16-bit registers serve as temporary storage and as the stack pointer (SP) and frame pointer (FP) in the stack manipulation instructions. The stack pointer contains the address of the top of the stack, i.e. the last word "pushed" onto the stack which is also the first word that may be "popped." The frame pointer contains the address of the highest location in a block of five words on the stack, a "frame," containing program status information used to return from a subroutine (see *Figure 3*).

The frame pointer is updated by the Save and Return instructions which are intended to be the first and last instructions, respectively, executed by a subroutine. When a Jump-to-Subroutine instruction is executed, the value PC+1 (and the value of the carry bit in 32K-word mode only) is stored in AC3. The Save instruction then pushes five key words onto the stack in the following order: first, the contents of AC0; second, the contents of AC1; third, the contents of AC2; fourth, the value of FP before the Save; and fifth, the contents of AC3. At this point, SP points to the top of the frame (which is the current top of the stack), and that address becomes the new value of FP. This new value of FP is also placed in AC3. When a Return instruction is executed, the five words stored in the frame referenced by FP are used to restore accumulators AC0 through AC2 to their values at the time preceding the Save. FP is restored to its previous value (pointing to the last previously saved five-word frame) and PC is loaded with the return address which had been placed in AC3 by the previous Jump-to-Subroutine and pushed onto the stack by the previous Save. The restored value of FP is also placed in AC3 by the Return instruction.

Information may also be moved between SP or FP and any of the four accumulators by the instructions MTFP, MFFP, MTSP, and MFSP without affecting the source register of the move or any of the registers not specified with the instruction. This allows setting up multiple stacks whose pointers are saved in main memory when not in use.

Addressing Ranges and Modes

The F9445 memory reference instructions support two address ranges and a variety of addressing modes. These modes include direct/indirect addressing which may be absolute, PC-relative, or indexed by AC2 or AC3. Additional addressing modes include auto-increment, auto-decrement, and address via stack and frame pointers. The two address ranges in which the F9445 can operate are 128K-byte (64K-word) or 64K-byte (32K-word) logical address space. The F9445 master resets to the 64K-byte (32K-word) address range. The 128K-byte (64K word) address range can be enabled or disabled under program control.

64K-Byte (32K-Word) Address Range

After the master reset is activated or the D64K instruction is executed, the F9445 operates in the 64K-byte (32K-word) address range. In this mode of operation, it uses 15-bit addresses to fetch up to 32K words from the memory and uses either the least-significant sixteenth bit to select high or low byte of the word in the byte instructions or the most-significant sixteenth bit to specify the remaining 15 bits of the word as an indirect address in multi-level indirect addressing instructions.

In the Load-Byte (LDB) and Store-Byte (STB) instructions, a 16-bit accumulator is specified as the byte pointer. The most significant 15 bits of the byte pointer are treated as the logical address of the word containing the byte which the least significant bit specifies, selecting the high (if "0") or low (if "1") byte of the word.

The remaining memory reference instructions specify effective addresses of 16-bit words via various (11) addressing modes described below.

Page Zero	In this mode the instruction provides an 8-bit absolute address to access the first 256 words (page zero) of memory.
PC Relative	In this mode the instruction provides an 8-bit twos-complement signed number which is added to the program counter to access 128 locations below and 127 locations above the address specified in the program counter.
Indexed by AC2 (or AC3)	In these two modes the instruction provides an 8-bit twos-complement signed number which is added to AC2 (or AC3) to access 128 locations below and 127 locations above the address specified in the accumulator.

The memory reference instruction may specify any of the above four memory addressing modes to be either direct or indirect. For direct addressing, the effective address computed using the eight address bits of the instruction is the final address of the target word to be stored or retrieved.

For indirect addressing, the effective address computed from the eight address bits of the instruction is used to fetch a 16-bit word that supplies the address of the target word. If the most significant bit of this word is "0", the 15 least significant bits provide the address of the target word. However, if the most significant bit of this word is "1", this specifies a further level of indirect address. In that case, the 15 least significant bits refer to the address of another word which could provide the final address of the target, depending on whether its most significant bit is "0" or "1". Thus, multiple levels of indirect addressing continue until a word is fetched with a most significant bit of "0". Such multiple levels of indirect addressing are only allowed in the 32K-word address range operations.

The next two types of addressing modes are the auto-increment and auto-decrement modes. When locations 20 through 27 (octal) are indirectly addressed, the auto-increment mode is activated: the contents of the specified location are first incremented and stored back and this new value is treated as the effective address (which can, in turn, be either direct or indirect). Locations 30 through 37 (octal) are used as auto-decrement locations in a similar manner.

The last type of addressing is stack addressing in which the address of the memory reference is derived from the stack pointer.

128K-Byte (64K-Word) Addressing Range

After the E64K instruction is executed, the F9445 starts operating with the 128K-byte (64K-word) addressing range. In this range, the F9445 uses 16-bit addresses to fetch up to 64K words from the memory and supports all the 11 addressing modes described previously. However, only one level of indirect addressing is allowed — the one specified in the instruction — since with 16-bit addresses there are no bits available in the words fetched to indicate further indirect addressing.

The byte pointer is also different in the 128K-byte (64K-word) case compared to the 64K-byte (32K-word) case. The 64K-word range byte pointer is 17 bits wide and is composed of the carry flag and the 16-bit accumulator specified in the LDB or STB instruction. The value of the least-significant bit of the 17-bit word selects the high (if "0") or low (if "1") byte of the word to be loaded or stored.

Instruction Set

The F9445 has fixed-length instructions, each of which is 16 bits long and divided into several fields. The fields are used to specify the operation code and other related actions, to define conditions and specify the CPU registers containing arguments, to define I/O device codes, and to provide the displacements for the calculation of effective addresses of memory locations.

The whole instruction set can be divided into five broad groups:

Memory Reference Instructions
Arithmetic-and-Logic Instructions
Stack Manipulation Instructions
I/O Instructions
Control Instructions

The Memory Reference instructions modify the contents of memory locations, alter program execution sequence, and move operands between the accumulators and memory locations. The contents of accumulators and the carry and overflow flags are processed by the Arithmetic-and-Logic instructions. The Stack instructions manipulate the registers and the memory in stack-associated operations. The I/O instructions effect data transfers between the accumulators and I/O devices. The Control instructions modify or interrogate the state of the CPU and operator console, performing such actions as controlling the status of the interrupt-on flag and reading the status of the console switch register.

Input/Output Operations

Input/output devices can transfer data to the F9445-based microcomputer via:

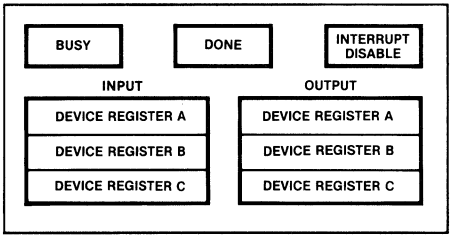
Programmed I/O using the I/O instructions of the F9445,

Memory-mapped I/O using the load/store instructions of the F9445, or

Direct memory access or data-channel transfers.

For programmed I/O, the device consists of up to three (minimum one) bidirectional 16-bit device registers, denoted as A, B, and C, and three 1-bit flags: Busy, Done and Interrupt Disable (see *Figure 4*). The 2-bit status word comprised of Busy and Done represents one of up to four possible states of the device, viz. idle, busy, partially done and completely done (refer to *Device Status Flags* subsection). The F9445 I/O instructions allow data transfers between any of the accumulators (AC0 through AC3) and any of the device registers (A through C), and can test and set the Busy, Done and Interrupt-Disable flags.

Fig. 4 I/O Device Model



The F9445 can transfer the contents of any accumulator to an I/O device by executing a Data-Out instruction. It can load data from an I/O device into any accumulator by executing a Data-In instruction. To test the status of an I/O device, the F9445 can execute a Skip-On-Status instruction.The I/O cycle has the same timing as the memory cycle (see *Figures 13* and *14*). Features of the I/O cycle are:

- 250 ns (at 24 MHz system clock) minimum cycle time
- Cycle time can be extended using RDYA, RDYD
- I/O instruction is output at address time
- $\overline{\text{STRBA}}$ is used to latch the I/O instruction
- $\overline{\text{STRBD}}$ is used to strobe the data
- O lines indicate the type of cycle as follows:

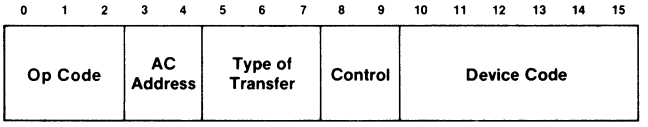
	$\overline{M}$	O_1	O_0	$\overline{W}$
I/O Input Execute	1	0	0	1
Instruction Fetch	0	0	0	1
I/O Output Execute	1	0	0	0
Interrupt Save	0	1	1	0

- The I/O devices can interrupt the normal flow of the program by using the common interrupt request line

Instruction Decode

An I/O instruction in the F9445 system comprises several fields as shown in *Figure 5*. This format accommodates data transfers between a CPU accumulator and any one of up to three bidirectional registers in any one of 62 I/O devices. Bits 10 through 15 are coded to represent device codes 00 through 76 (octal). The all "1s" device code, 77 octal, is reserved for CPU control instructions and should not be assigned to any unique I/O device; for similar reasons, device code 1 is also reserved; by convention, device code 0 is not used.

Fig. 5 Input/Output Instruction Fields



Bits 3 and 4 specify the address of any accumulator involved in an I/O instruction. When no accumulator is involved, both bits are ignored. The function bits 5, 6, and 7 define the I/O operation to be performed. Bits 8 and 9 control or test the status of the device busy and done flags.

The eight standard I/O instructions were listed previously in the instruction set description of the introduction to this section. The No-Input/Output (NIO) instruction is a "no data transfer" instruction that can be used to set the busy and done flags as required, by attaching the appropriate flag-setting mnemonic. The F9445 executes a "dummy" data out transfer. The status of a device's busy and done flags is tested by executing a Skip (SKP) instruction that causes a specific I/O device to put its busy and done flag states on lines IB_0 and IB_1 of the common information bus. If the flag state satisfies the condition specified by the busy/done flag-testing mnemonic appended to SKP, the CPU skips the next instruction. The remaining six standard I/O instructions first move data between an accumulator and any one of the device registers A, B, or C. After the transfer is completed, the busy/done flags are set as specified in the I/O instruction.

There are three I/O instructions that are common to all I/O devices: Interrupt-Acknowledge, Mask-Out, and Clear-I/O-Devices. The device code for these three instructions is 77 (octal).

When the F9445 executes the I/O instruction, the $\overline{M}$ and O lines will indicate an I/O operation ("100"). The O lines are valid on the rising edge of SYN. The device address (bits 10-15) must be decoded by each device on the I/O bus. Transfers of information to and from the F9445 are timed with $\overline{STRBD}$ in the same way as the memory cycle.

At the address time, the F9445 outputs the I/O instruction on the information bus. This can be used to generate I/O signals on systems without an I/O controller. $\overline{STRBA}$ is generated and can be used to latch the I/O instruction externally. The interrupt-disable, busy and done flags organize interrupt-driven program-controlled I/O operations. The CPU controls the interrupt-disable flag. Both the CPU and the device can control the busy and done flags.

Device Status Flags

Interrupts from a device are disabled when the interrupt-disable flag of the device is set to "1". Interrupts are enabled when the flag is clear. Interrupt requests are generated whenever the device sets the done flag.

During programmed I/O, the interrupt-disable flag is normally set to disable interrupts, and the busy and done flags define the status of the device for the CPU. The busy and done flag states are coded to represent the indicated device conditions, as follows.

Busy	Done	Device State
0	0	Device idle
1	0	Device busy
0	1	Device completely done
1	1	Device partially done

The sequence of I/O transactions is normally dictated by the speed at which the device can communicate with the CPU. If the CPU operates at a higher speed than a device, it enters a wait loop between each I/O transaction with the device. During execution of the loop, the CPU repeatedly monitors the busy or done flag to determine when the device is ready for the next I/O operation.

During an output operation, one instruction stores data in the desired device register and places the device in the busy state. The CPU then enters a wait loop which terminates when the device has cleared busy and set done to signal readiness for the next output operation.

To initiate an input transaction, the device sets the done flag. One instruction reads data from the appropriate device register and places the device in the busy state. The CPU then enters a wait loop which terminates when the device has cleared busy and set done to indicate that it has the next data ready.

Interrupts

The interrupt request, $\overline{\text{INTREQ}}$, line is common to all I/O devices. When the device completes an I/O operation, it should set the done flag. Concurrently, if the device is enabled to interrupt, it should assert the active LOW on the $\overline{\text{INTREQ}}$ line. The processor responds to the interrupt request after completing execution of the current instruction. It then clears the interrupt-on flag so no further interrupts can be started, saves PC (which points to the next instruction) in location 0, and executes a "jump-indirect-to-location-1" instruction to jump to the interrupt service routine. Location 1 should contain the address of the interrupt routine or an indirect address to the routine. The F9445, when interrupted, can check for the source of the interrupt in two ways:

It can test the state of the done flags in the various devices, one by one, by executing Skip-on-Done instructions; or

It can test the state of the I/O devices by executing the Interrupt-Acknowledge instruction, causing the device that had sent an interrupt request to respond by placing its device code on bits 10 through 15 of the information bus.

As several devices can request interrupt simultaneously, device priority may be established in a daisy-chain fashion by a physical connection of a serially propagated signal, Interrupt Priority. The first device requesting an interrupt and having its Interrupt-Priority-In line HIGH has priority, and it answers the Interrupt-Acknowledge instruction, at the same time blocking the propagation of the interrupt-priority signal by putting its Interrupt-Priority-Out line in a LOW state.

The interrupt-priority signal is generated in the device having the highest priority. The F9445 can disable the interrupt system in each I/O device by placing a mask on the information bus while executing the Mask-Out instruction.

Each bit in the mask is assigned to a specific device. When that bit is "1", the interrupt system is disabled. A "0" in that bit enables the device.

After servicing a device, the routine should restore the pre-interrupt states of the accumulators and carry, turn on the interrupt, and jump to the interrupted program. The instruction that enables the interrupt sets interrupt on (INTON), but the flag has no effect until the next instructior begins. Thus, after the instruction that turns the interrupt back on, the processor always executes one more instruction (assumed to be the return to the interrupted program) before another interrupt service can start. If the service routine allows interrupts by higher priority devices, the routine should turn off the interrupt, before dismissing as indicated above, to prevent further interrupts during dismissal. In dismissing, the routine should re-enable lower priority devices.

The interrupt request input $\overline{\text{INTREQ}}$ is negative-level sensitive and is synchronized in the processor. Externally, interrupt requests may be latched with the leading edge of SYN. The interrupt request may be reset by the external I/O controller from a decode of the I/O instruction INTA.

The F9445 recognizes two other types of interrupts:

Abort Interrupt — This is activated by the active LOW of the $\overline{\text{ABORT}}$ input. The processor responds by:

Aborting the instruction being executed,
Storing the address of the aborted instruction in location 46 (octal), and
Jumping indirect to location 47 (octal).

Stack Overflow interrupt—This is an internal interrupt caused when the stack overflows; i.e., when a stack operation (PSHA, PSHF, PSHR, SAVE, TOPW) writes over a page boundary (mod 256). This interrupt is of higher priority than the external interrupt ($\overline{\text{INTREQ}}$); the processor responds, at completion of the current instruction by:

Clearing the interrupt-on flag (to "0"),
Storing the updated program counter in location 0, and
Jumping indirect to location 3 (octal).

The interrupt-save cycle follows the interrupt. It can be externally detected by the code "011" on the O lines and used, for example, to switch an external mapper to non-mapped mode.

The order of priority of requests and interrupts, from highest to lowest, is as follows: $\overline{\text{MR}}$, $\overline{\text{ABORT}}$, $\overline{\text{DCHREQ}}$, Stack Overflow Interrupt, $\overline{\text{INTREQ}}$, and $\overline{\text{CONREQ}}$.

Data Channel
The data channel has three methods of operation with the F9445:

Data-channel cycle with F9445 controlling the memory,
Data-channel cycle with external memory control, and
Autonomous-bus cycle using bus arbitration scheme.

The sequence of events during a data-channel cycle is as follows:

1. $\overline{\text{DCHREQ}}$ is set.
2. F9445 responds by setting $\overline{M}$, O_1, and O_0 to "101" and $\overline{\text{BUSREQ}}$ to "1". This is recognized externally as Data-Channel Acknowledge and can be used to reset $\overline{\text{DCHREQ}}$ if it is the last data-channel cycle required.
3. F9445 3-states the bus and sends $\overline{\text{STRBA}}$.
4. The external logic must supply an address at this time. The address time can be extended with RDYA.
5. F9445 outputs $\overline{\text{STRBD}}$.
6. The controller transmits or receives the data-channel data and responds with RDYD, concluding the cycle.

Console Operation
Console operation allows examination and modification of the F9445 internal registers without executing programs in main memory. This is very useful for system diagnostics even when the memory or I/O part of the microcomputer system is not fully functional.

Upon request for console operation, the processor will execute one of a number of console operations depending on a console code on the information bus (see *Figure 6*). This facilitates the connection of an external console for monitoring and test purposes. The following sequence is used to execute a console operation:

1. $\overline{\text{CONREQ}}$ is set LOW.
2. The processor finishes the current instruction.
3. The processor sets the $\overline{M}$ and O lines to "110" (console code in).
4. In response to the $\overline{M}$ and O lines being set to "110", the console logic supplies a code on the information bus corresponding to the desired operation, which is selected onto the bus with $\overline{\text{STRBD}}$.
5. The console logic resets $\overline{\text{CONREQ}}$.
6. The processor executes the console operation.
7. The processor may read or write data from the console switches or console lamps. In this case, the $\overline{M}$ and O lines are set to "111" (console data). In most cases, the processor halts after the console operation by entering a Wait state. The exceptions are Continue and APL.

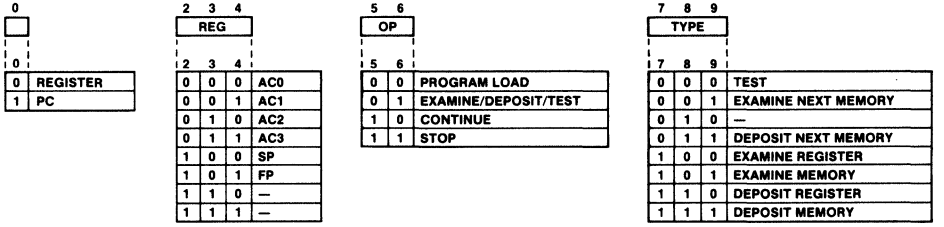
Console logic can be implemented in three levels of simplicity:

No Console Code — If a $\overline{\text{CONREQ}}$ is generated and no console code supplied, the default bus value ("0") will cause the processor to execute APL. This sets the PC to −1, then starts normal execution. This is the minimal console operation required.

Limited Console Operation — A subset of operations can be arranged with a 2-bit console code. These operations are APL, Test, Continue, and Halt.

Full Console Operation — A 9-bit code (see *Figure 6*) defines the full set of console operations. Single-Step is not implemented directly, but can be arranged using Continue first, the Continue operation is specified; after the first instruction is fetched, a new $\overline{\text{CONREQ}}$ is generated and the operation is changed to Halt.

Fig. 6 Console Codes



0	REGISTER
1	PC

2	3	4	REG
0	0	0	AC0
0	0	1	AC1
0	1	0	AC2
0	1	1	AC3
1	0	0	SP
1	0	1	FP
1	1	0	—
1	1	1	—

5	6	OP
0	0	PROGRAM LOAD
0	1	EXAMINE/DEPOSIT/TEST
1	0	CONTINUE
1	1	STOP

7	8	9	TYPE
0	0	0	TEST
0	0	1	EXAMINE NEXT MEMORY
0	1	0	—
0	1	1	DEPOSIT NEXT MEMORY
1	0	0	EXAMINE REGISTER
1	0	1	EXAMINE MEMORY
1	1	0	DEPOSIT REGISTER
1	1	1	DEPOSIT MEMORY

Bus Arbitration

The F9445 contains three signals that allow more than one processor to share a common bus:

$\overline{\text{BUSREQ}}$—This is LOW at the beginning of every cycle in which the F9445 requires use of the bus.

BUSGNT—When LOW, it is used to halt the processor indicating the bus is unavailable.

$\overline{\text{BUSLOCK}}$—This indicates that the current bus cycle and the following bus cycle from the processor must not be interrupted by a cycle from another processor.

The $\overline{\text{BUSLOCK}}$ signal has two purposes. One purpose is to prevent the external memory address register from being overwritten during those instructions that rely on the address remaining in this register. The other purpose is to provide a method of synchronizing separate software tasks using a standard semaphore system. An external arbiter is required to determine which processor has access to the bus.

Applications

Static Memory Interface

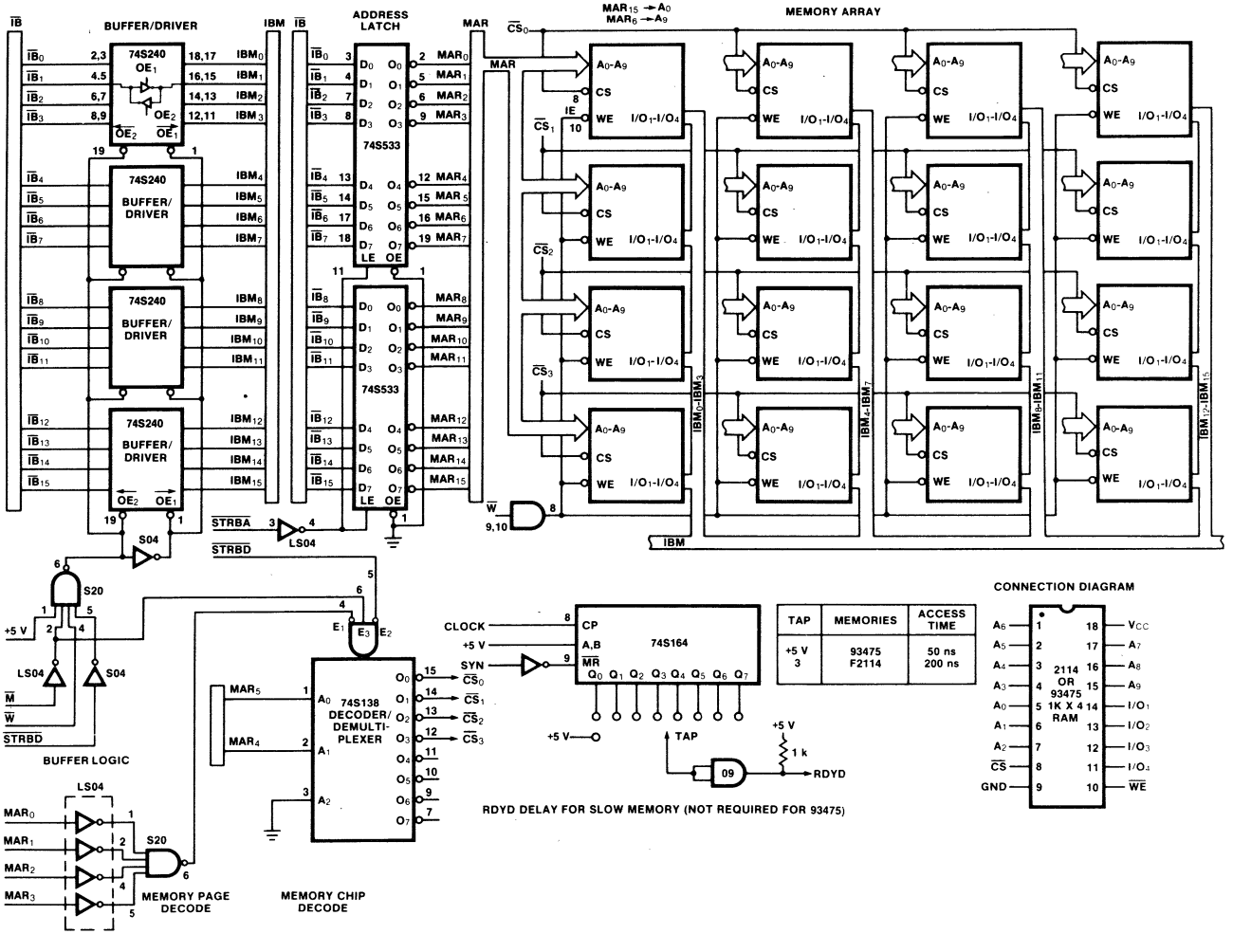
The F9445 bus structure allows easy connection of static memory. Both address and data are multiplexed onto the 16-bit information bus $\overline{\text{IB}}_{(0\text{-}15)}$. The mutually exclusive signals $\overline{\text{STRBA}}$ and $\overline{\text{STRBD}}$ indicate that the information bus is carrying address or data, respectively. The $\overline{\text{M}}$ signal ($\overline{\text{M}}$ = LOW) indicates that a memory cycle is taking place on the bus, while the $\overline{\text{W}}$ signal indicates whether the operation is a read or write. The timing of the $\overline{\text{STRBD}}$ is shorter for a write operation, to allow positive hold time for the memories. The signal RDYD may be held LOW to stretch the memory cycles for slow memories.

A typical scheme is shown in *Figure 7*. This diagram shows a 4K × 16 static RAM configuration (2114-type 1K × 4). The bus is buffered by a 74240 inverting 3-state buffer. Buffering is optional and depends on fan-out requirements of the memories. The buffer is normally connected for output, but is connected for input when necessary by a simple decode of the $\overline{\text{M}}$, $\overline{\text{W}}$ and $\overline{\text{STRBD}}$ lines.

An address latch (74533) is clocked with $\overline{\text{STRBA}}$. The memory address is decoded from these outputs and forms the chip-select ($\overline{\text{CS}}$) inputs to the RAM.

A shift register (74164) provides a time delay for RDYD for slow memories. An alternative for this would be a one-shot (9602). Fast memories do not require RDYD delayed.

Fig. 7 Static Memory Connection Scheme



TAP	MEMORIES	ACCESS TIME
+5 V 3	93475 F2114	50 ns 200 ns

Input/Output

The F9445 I/O can utilize a simple scheme similar to the memory connection. To take full advantage of standard F9445 I/O instructions, however, I/O instructions must be externally decoded. An F9445 support circuit (F9448, F9447, F9470) can be used for this purpose.

To implement standard F9445 I/O without the support circuits mentioned above requires external logic. This can be implemented with an FPLA (93459). *Table 1* illustrates the PLA for I/O.

Table 1 I/O PLA Listing

```
                                  *A LLLLLLLL
*P 00 *I --L-LLLHLHL--HLL *F -----AAA
*P 01 *I --L-HLLHLHL--LHL *F ------AA
*P 02 *I --L-LLLLHHL--HLL *F -----A-A
*P 03 *I -HH------L------ *F -----A-A
*P 04 *I --L-LLLLHHL--LHL *F -------A
*P 05 *I -LH------L------ *F -------A
*P 06 *I --L-LLLLHHL--LLH *F -----AA-
*P 07 *I --L-LLLLHHL--LHH *F ------A-
*P 08 *I --L-LLLLLHL--LHL *F -----A--
*P 09 *I ---------H---HHH *F A-------
*P 10 *I --L-LLLHLHL--HHH *F --AAA---
*P 11 *I --L-HLLHLHL--HHH *F ---AA---
*P 12 *I --L-LLLLHHL--HHH *F --A-A---
*P 13 *I --L-LLLLHHL--HHL *F --A-A---
*P 14 *I --LHLLLHLHLLH--- *F -AAAA---
*P 15 *I --LHHLLHLHLLH--- *F -A-AA---
*P 16 *I --LHLLLLHHLLH--- *F -AA-A---
*P 17 *I --LHLLLLHHLHH--- *F -A--A---
*P 18 *I --LHLLLHLHLHL--- *F -AAA----
*P 19 *I --LHHLLHLHLHL--- *F -A-A----
*P 20 *I --LHLLLLHHLHL--- *F -AA-----
*P 21 *I --L-HHHHHHH--HLH *F --AA----
*P 22 *I --L-HHHHHHH--HHL *F ---A----
*P 23 *I --L-HHHHHHH--LLH *F --A-----
*P 24 *I --L-HHHHHHH----- *F A-------
```

Key for Table 1:

- *A = Active level of outputs
- *P = Product term number
- *I = Inputs
- *F = Outputs
- – = Don't care
- H = High level
- L = Low level
- A = Active

The schematic (see *Figure 8*) shows a UART connection. The FPLA decodes the instructions and produces outputs from three multiplexers (74138). Spare outputs on these multiplexers can be used to drive other I/O devices.

Busy, done, mask and interrupt latches for both input and output are implemented. The baud rate generator (4702) is programmable for baud rates from 110 to 9600 baud.

In this scheme, the I/O bus is buffered (74240); this is optional. A one-shot (9602) provides a processor cycle delay by holding RDYD low. This allows the use of a slow UART (TR1263B). Converters (1488, 1489) are used for RS232-level connection, and current loop drivers are switch selected as shown. A one-shot (9602) provides a pulse for a TTY reader delay.

Dynamic Memory Control

Since dynamic memory is more difficult than static memory to connect to any processor, the F9445 requires some additional circuitry to drive dynamic memories (see *Figure 9*). There are several approaches to dynamic memory control:

Using an LSI special-purpose dynamic memory controller (e.g. F9446), which is by far the simplest solution;

Using standard SSI or MSI for the controller, requiring considerable board area;

Using software-assisted techniques, which reduces hardware requirements but can result in poorer overall performance; or

Using a standard MSI dynamic memory controller (e.g. 9642) with additional timing and control logic.

The last alternative has the advantage of using standard parts with a low part count and no software overhead. This is the scheme shown in *Figure 9*. The memory address register, data buffer and address decoder are required for any memory, static or dynamic. The 9642 multiplexes the 14-bit address for the dynamic memories, seven bits at a time. The memories require two strobes: a row address strobe ($\overline{RAS}$) and a column address strobe ($\overline{CAS}$). In the scheme shown, all memory chips receive the same $\overline{CAS}$ strobe, but the $\overline{RAS}$ strobe depends on the address. The strobes are sequenced using a combination of F9445 timing signals ($\overline{STRBA}$, SYN, $\overline{STRBD}$) and other signals generated by a 74164 shift register.

Fig. 8 F9445 Input/Output Connection Scheme (1 of 2)

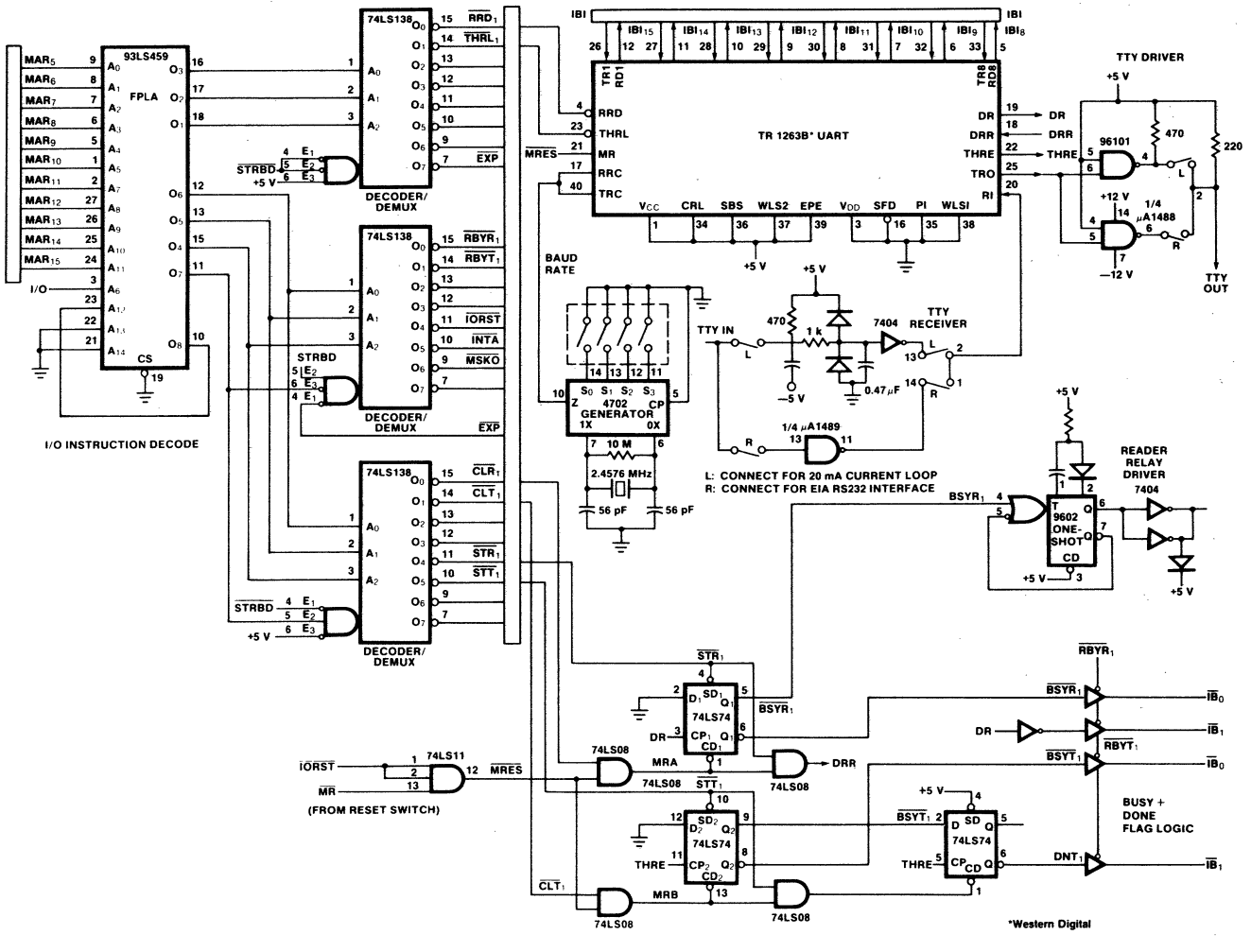


Fig. 8 F9445 Input/Output Connection Scheme (2 of 2)

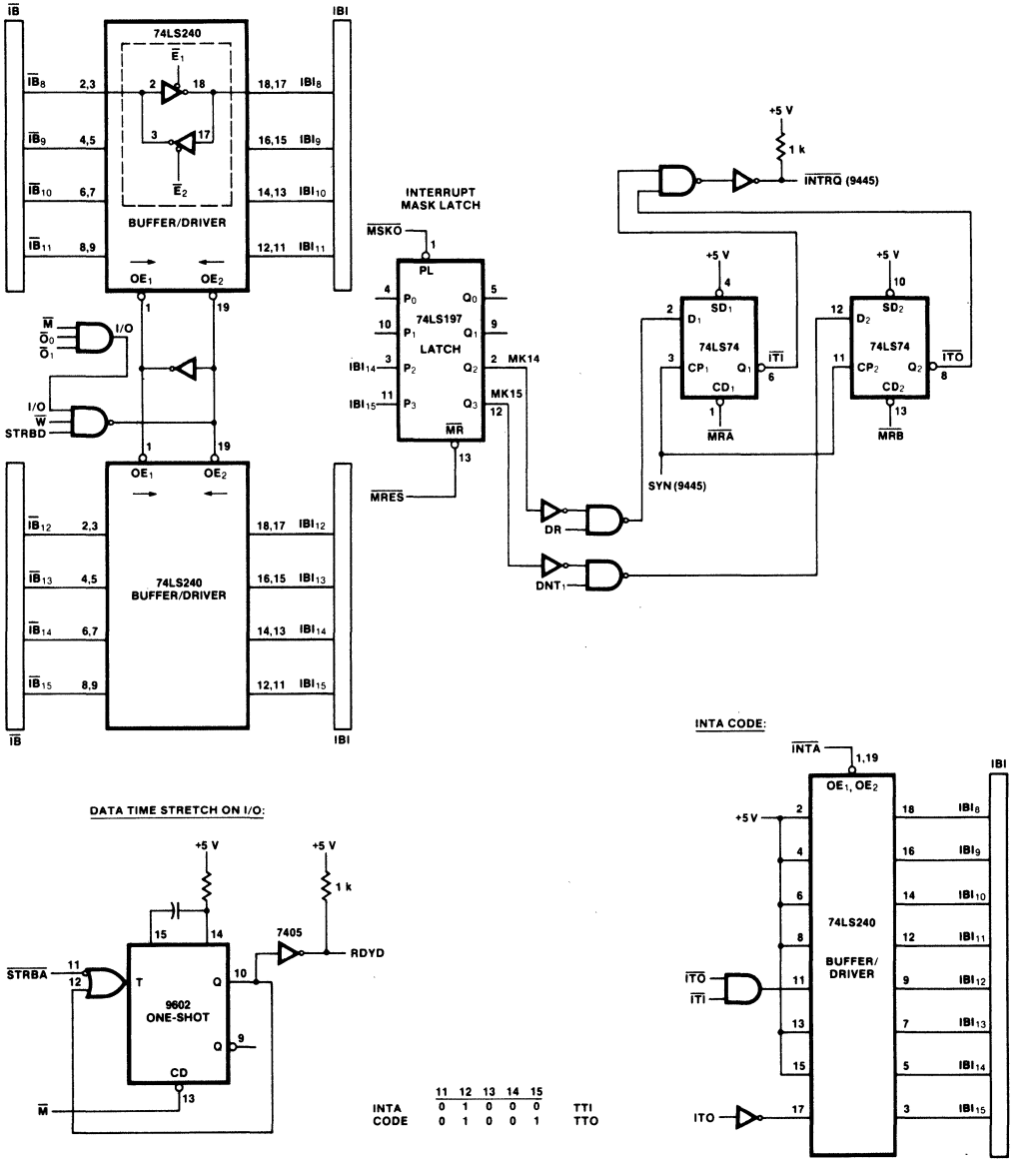
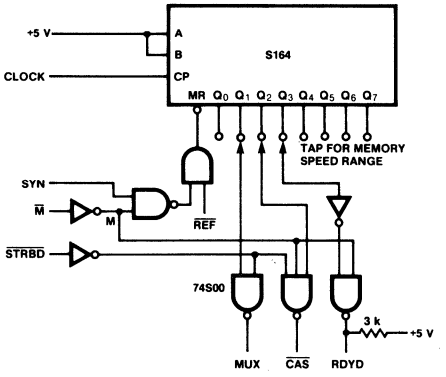
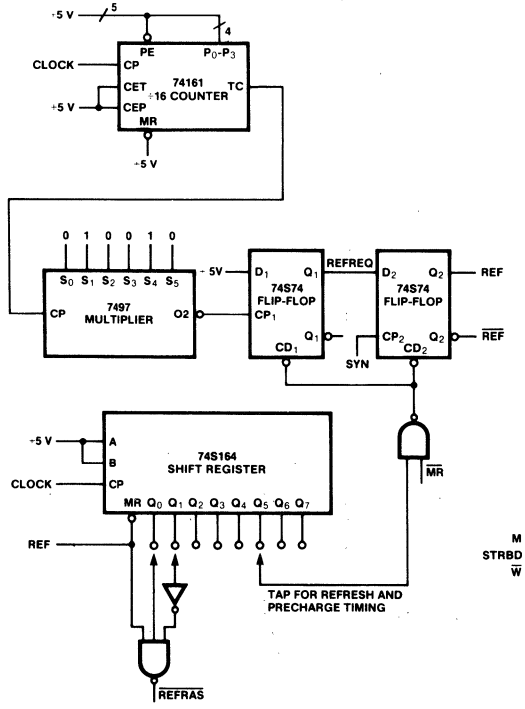


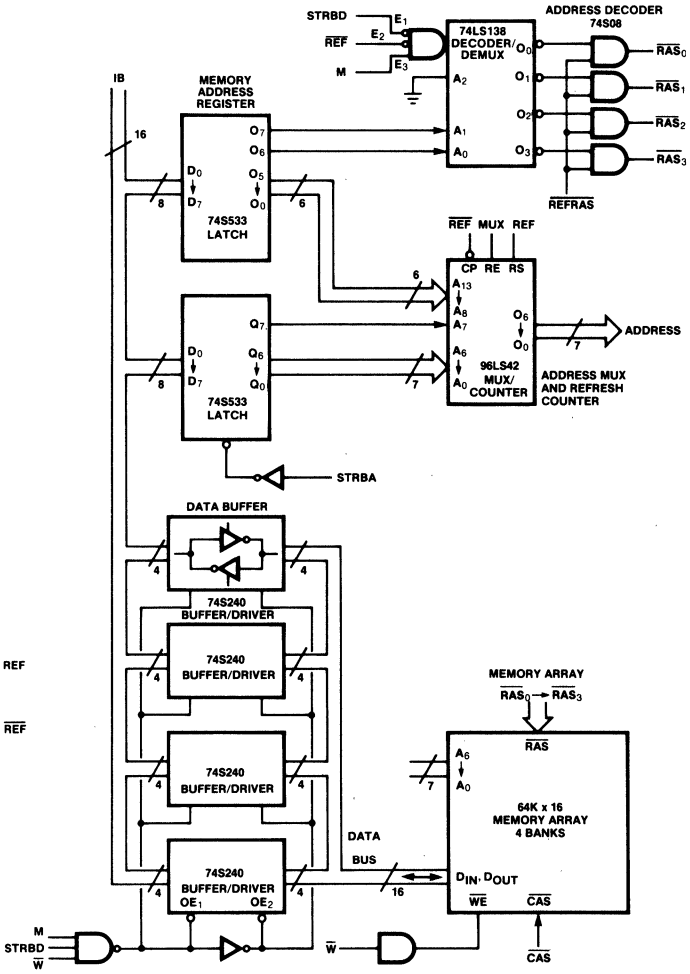
Fig. 9 F9445 Dynamic Memory Connection Scheme

Dynamic Memory Timing



Refresh Timing





Notes

1. D_{IN} connected to D_{OUT} connected to Data Bus.
2. All F16K devices have the same address lines and $\overline{CAS}$, $\overline{WE}$ line.
3. Each bank of 16 F16K has a separate $\overline{RAS}$ line (4 banks).
4. Each slice of 4 F16K is connected to a separate data bus line (16 slices).

The memory requires "refreshing" every 2 ms. The 9642 contains a 7-bit refresh counter. Every 15.63 ms (2/128), the memory controller enters "refresh" mode. This is synchronized with SYN to avoid any conflict. Another 74164 shift register controls the refresh timing which requires only an $\overline{RAS}$ strobe. After the refresh cycle, the refresh counter is incremented and the normal memory timing is resumed.

The refresh cycle takes place when needed and may take place during non-memory processor cycles. In these cases, the processor is not halted, and the refresh cycle is overlapped.

Different memory types have different speed requirements. These requirements can be met by changing the "taps" on the 74164 shift registers.

Console Control

On an application board, a minimal console is usually required. The APL (automatic program load) function can be easily implemented by pulsing the Console Request line LOW. There are no critical timing requirements since this signal is latched internally. The F9445 will continue to execute APL commands until the Console Request is raised. Since the bus must be HIGH for the APL to execute correctly, bits 5 and 6 of the bus may be tied to +5 V through 3 kΩ resistors as pullups.

For debugging and evaluation purposes, a console is a very useful tool. It gives complete control of the processor independent of software and memory operation.

Since the console commands are microprogrammed into the F9445, a full console design is fairly simple, the simplest full console uses the F9470 console-controller circuit, which drives an RS232 terminal and contains two serial I/O ports and a timer. The F9447 I/O controller can also be used to provide some console functions.

Interfacing to standard switches and lamps requires switch debouncing and encoding operations. The circuit shown in *Figure 10* uses R-S latches for switch debouncing and an FPLA (93409) for switch encoding.

An address latch is strobed on every $\overline{STRBA}$, and a data latch is strobed on every $\overline{STRBD}$ except "console code read." This results in the correct display on the lamps. The data switches are enabled with "console data read."

A Single-Step function is included. This function requires two additional latches, plus some decode logic, and implements a Continue followed by a Halt.

The Console Request is set whenever any operation switch is pressed and is reset when the console code is read from the FPLA. The circuit provides for control of two processors sharing the same bus.

All the switches are momentary-action type except the data switches and the select-processor switch.

A full listing of the FPLA is shown in *Table 2*.

The console provides all F9445 console functions, including Self-Test, plus the additional function of Single-Step, and is compact enough to be implemented with all switches, lamps, logic and connectors on a double-sided 17½-by-5½-inch printed-circuit board.

Table 2 Console PLA Listing

```
                               *A LLLLLLLL
*P 00 *I -H-HHHHHHHHHL---      *F -----A--
*P 01 *I -H-HHHHHHHHL----      *F --A--A--
*P 02 *I -H-HHHHHHHL-----      *F --AA-A--
*P 03 *I -H-HHHHHHL---H--      *F ----AA--
*P 04 *I -H-HHHHHHL---L--      *F ----AA-A
*P 05 *I -H-HHHHHL-------      *F --A-AA--
*P 06 *I -H-HHHHL-----H--      *F ---AAA--
*P 07 *I HL-H---L-----H--      *F ---AAA--
*P 08 *I HL-H---L-----L--      *F ---AAA-A
*P 09 *I HL-H-----L---L--      *F ----AA-A
*P 10 *I HL-H-----L---H--      *F ----AA--
*P 11 *I -H-HHHHL-----L--      *F ---AAA-A
*P 12 *I -H-HHHL---------      *F --AAAA--
*P 13 *I -H-HHL----------      *F --------
*P 14 *I -H-HL-----------      *F ------A-
*P 15 *I ---L------------      *F -----AA-
*P 16 *I HL-H------------      *F ------A-
*P 17 *I LL-H-----H------      *F -----AA-
*P 18 *I -H-HHHHHHHHHH---      *F -----AA-
```

Key for Table 2:

- *A = Active level of outputs
- *P = Product term number
- *I = Inputs
- *F = Outputs
- – = Don't care
- H = High level
- L = Low level
- A = Active

6

Fig. 10 F9445 Console Connection Scheme (1 of 3)

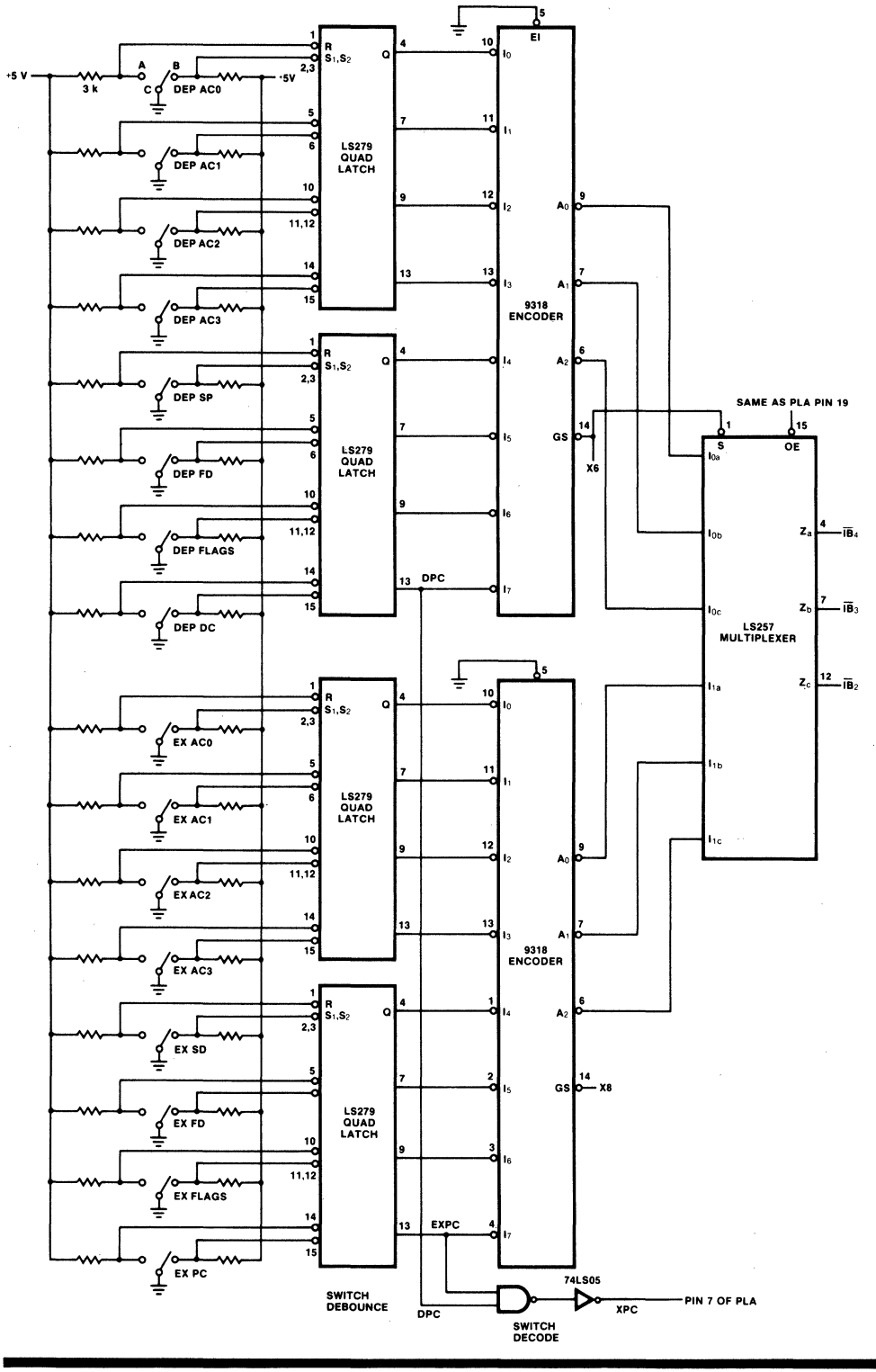


Fig. 10 F9445 Console Connection Scheme (2 of 3)

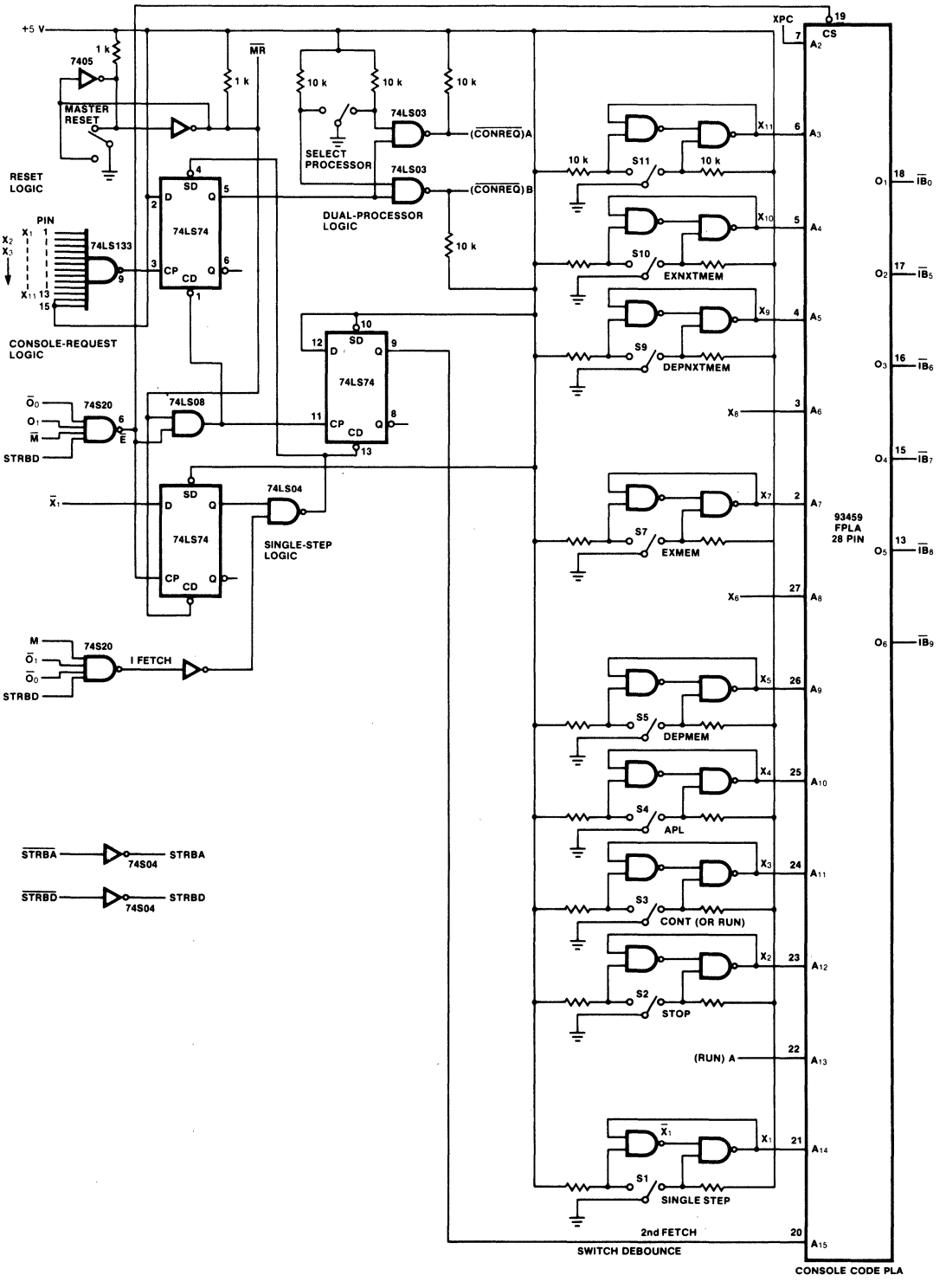
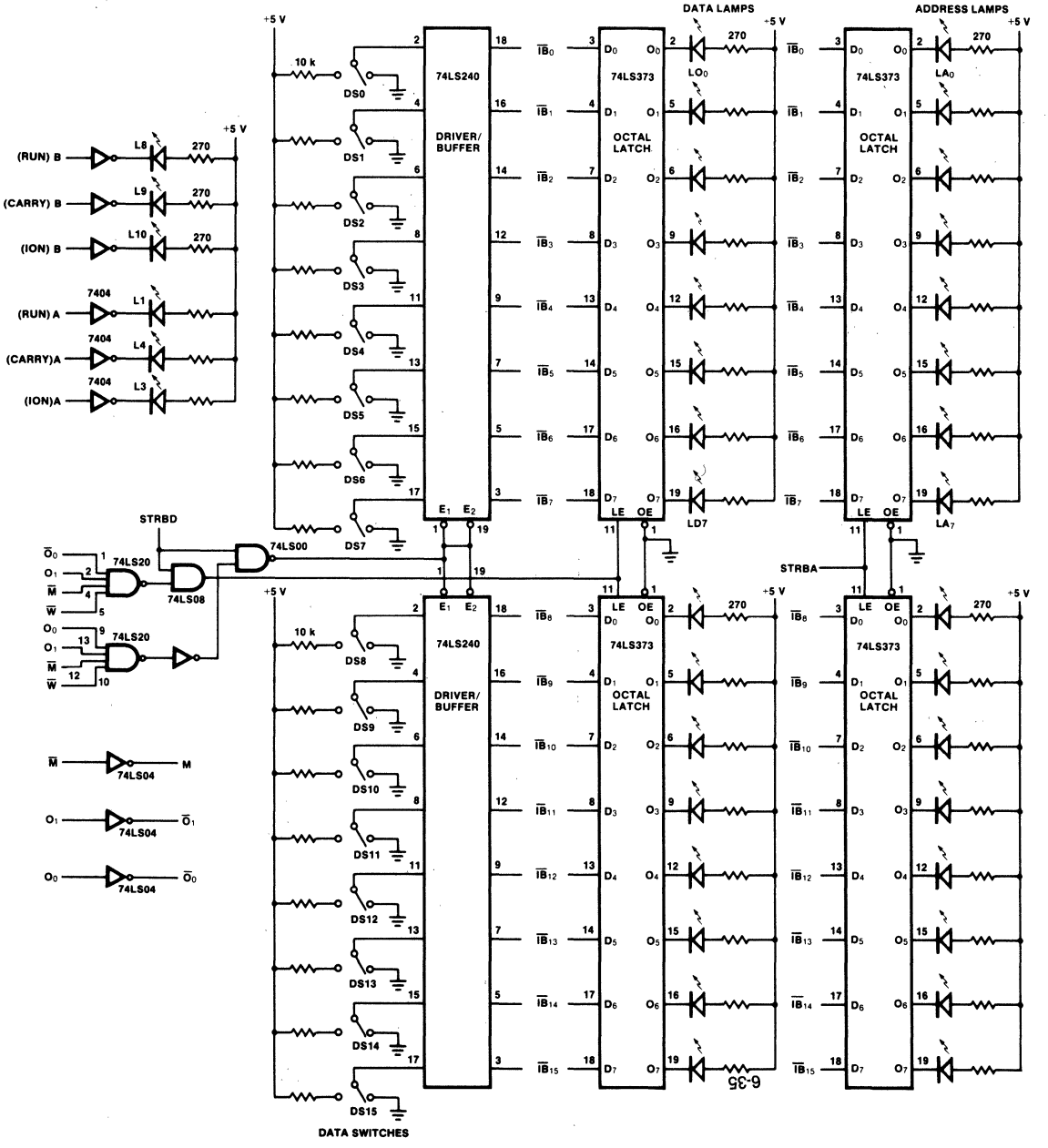


Fig. 10 F9445 Console Connection Scheme (3 of 3)



A Multiprocessor Scheme

There are many ways to envision two or more processors working concurrently. The method of interconnecting the processors depends on the application and the performance objectives. Listed here are a few of the options.

Independent Operation

For those processors which can be made to run independent tasks, this provides the most efficient scheme. Each processor has independent memory and resources.

Shared I/O

Each processor has its own memory but shares an I/O bus. This allows high-speed operation while minimizing system resource requirements.

Local and Common Memory

This gives a good compromise between performance and resource requirements. Each processor normally runs from its own memory at high speed. Accesses to a common memory are rarer and, because of the arbitration problems, slower.

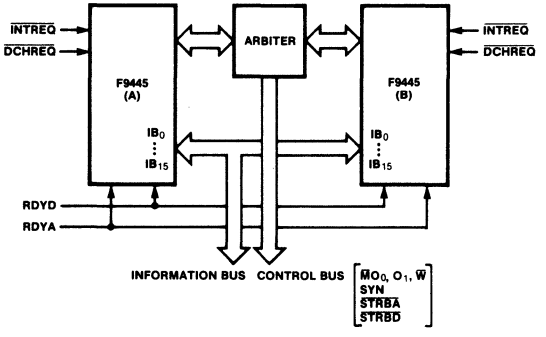
Tightly Coupled

Two or more processors share the same memory and I/O. This scheme is easiest to implement but, because of the completely shared resources, does not give as high performance as the Local and Common Memory scheme. However, for certain applications and for two processors only, this scheme can give a considerable performance increase over a single-processor system with very little hardware overhead. This scheme is described in the following paragraphs.

A general scheme for two tightly coupled F9445 processors is shown in Figure 11. The processors share a common bus and an arbiter selects which processor uses the bus and multiplexes the control lines accordingly. The I/O arbitration scheme is very simple: each processor assigns the bus to the other processor when it commences any cycle that does not use the bus, as long as $\overline{\text{BUSLOCK}}$ is not set.

The scheme is most efficient when the instruction mix includes many "long" instructions, such as Multiply, Divide, Parametric Shift and Normalize. Since only one processor is using the bus at any time, the synchronization signals RDYA and RDYD can be the same for both processors. However, the interrupt request ($\overline{\text{INTREQ}}$) and data-channel request ($\overline{\text{DCHREQ}}$) lines should be separate to avoid any conflicts in I/O handling.

Fig. 11 A Possible General Multiprocessor Scheme



F9445 Instruction Execution Times

Instruction	Clock Cycles	Execution Times			Notes
		16 MHz	20 MHz	24 MHz	
COM	6	0.375	0.3	0.25	
NEG	6	0.375	0.3	0.25	
MOV	6	0.375	0.3	0.25	
INC	6	0.375	0.3	0.25	Times for no-skip or unfulfilled skip;
ADC	6	0.375	0.3	0.25	for fulfilled skip; add 0.3 (0.25 at
SUB	6	0.375	0.3	0.25	24 MHz)
ADD	6	0.375	0.3	0.25	
AND	6	0.375	0.3	0.25	
OR	6	0.375	0.3	0.25	
MUL	70	4.375	3.5	2.9	
MULS	70	4.375	3.5	2.9	
DIV (Normal)	86	5.375	4.3	3.6	
DIV (Overflow)	14	0.875	0.7	0.58	
DIVS (Normal)	114	7.125	5.7	4.7	
DIVS (Overflow)	26	1.625	1.3	1.1	
NORM	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	n = number of steps needed for normalization. Time is 0.7 (0.59) if n=0.
SLLD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	n = number of shifts;
SALD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	time is 0.7 (0.59) if n=0.
SARD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	
SLRD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	
SKNV	14	0.875	0.7	0.58	
JMP	6	0.375	0.3	0.25	Times for page-zero addressing; add
JSR	6	0.375	0.3	0.25	0.3 (0.25) for indirect; add 0.3 (0.25)
ISZ	22	1.375	1.1	0.92	for auto-increment/decrement; add
DSZ	22	1.375	1.1	0.92	0.2 (0.18) for indexed.
LDA	12	0.75	0.6	0.5	
STA	12	0.75	0.6	0.5	
LDB	24	1.5	1.2	1.0	
STB	26	1.625	1.3	1.1	
PSHA	16	1.0	0.8	0.67	
POPA	16	1.0	0.8	0.67	
PSHF	16	1.0	0.8	0.67	
POPF	16	1.0	0.8	0.67	
POPJ	16	1.0	0.8	0.67	
PSHR	16	1.0	0.8	0.67	
TOPR	16	1.0	0.8	0.67	
TOPW	16	1.0	0.8	0.67	
MTSP	6	0.375	0.3	0.25	
MTFP	6	0.375	0.3	0.25	
MFSP	6	0.375	0.3	0.25	
MFFP	6	0.375	0.3	0.25	
SAV	60	3.75	3.0	2.5	
RET	80	5.0	4.0	3.3	
DSP	6	0.375	0.3	0.25	
NIO	12	0.625	0.6	0.5	
SKP	16	1.0	0.8	0.67	
DIA/B/C	12	1.0	0.6	0.5	
DOA/B/C	12	1.0	0.6	0.5	
ETRP	10	0.625	0.5	0.42	
DTRP	10	0.625	0.5	0.42	
E64K	10	0.625	0.5	0.42	
D64K	10	0.625	0.5	0.42	

Note: Execution times are given for 20 MHz and 24 MHz clock. The clock may be operated from > 0 to 24 MHz within the specified temperature and voltage range.

Fig. 12 ALU Cycle Timing*

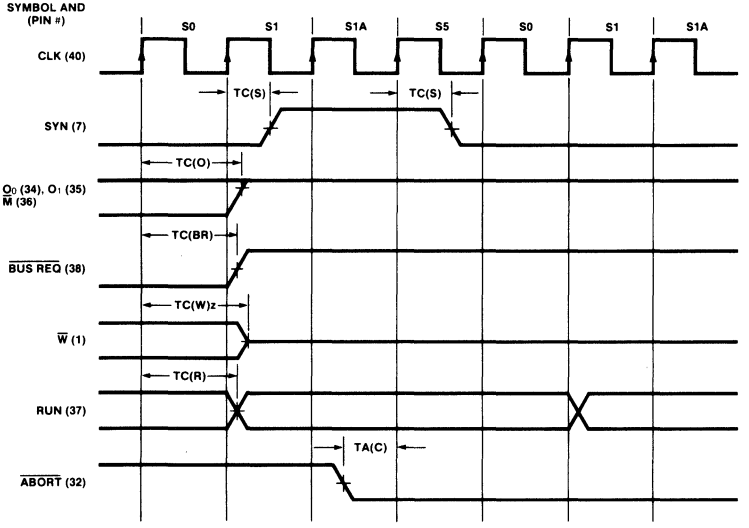
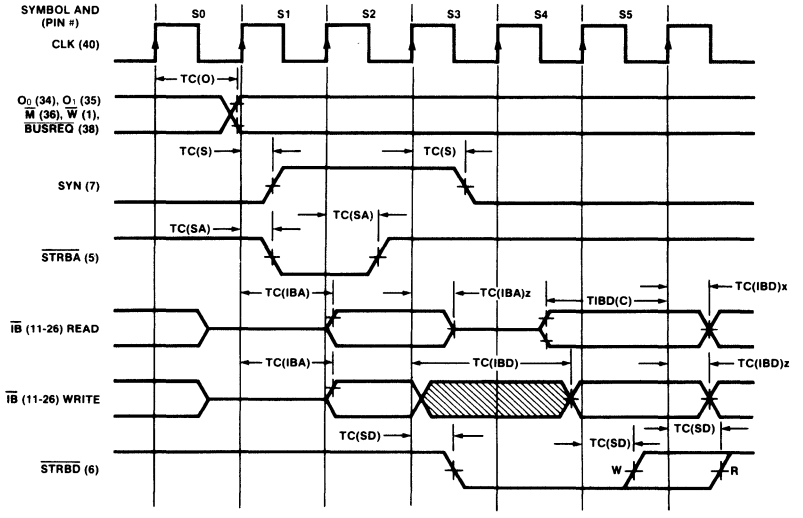
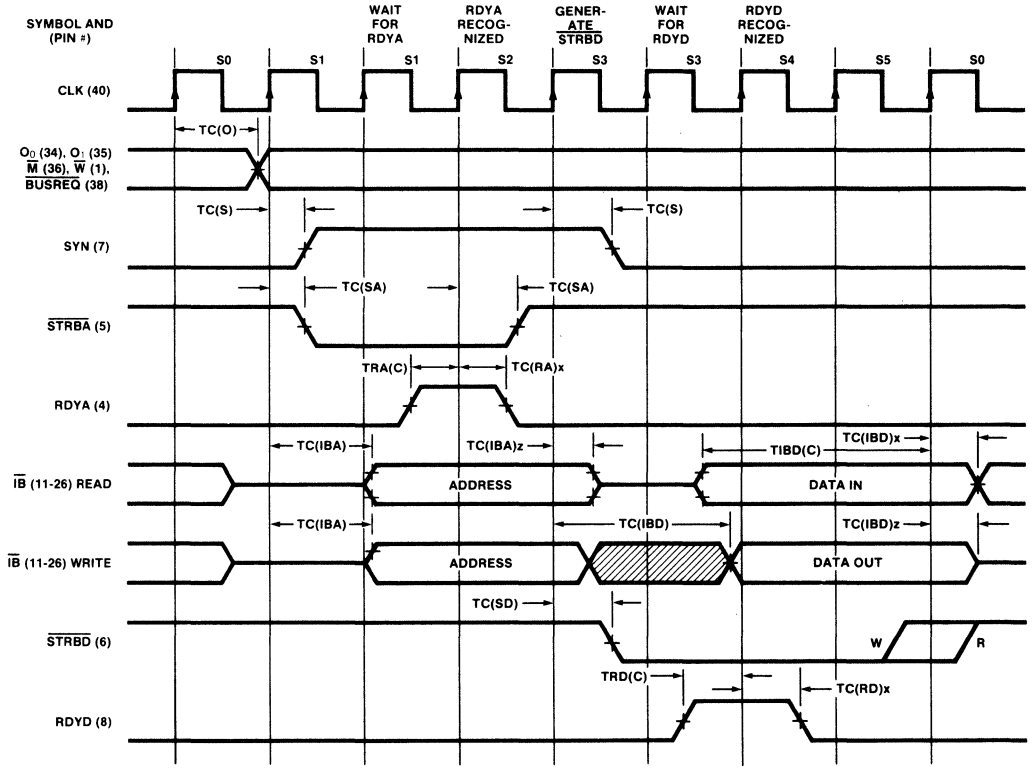


Fig. 13 Minimum Memory Cycle Timing*



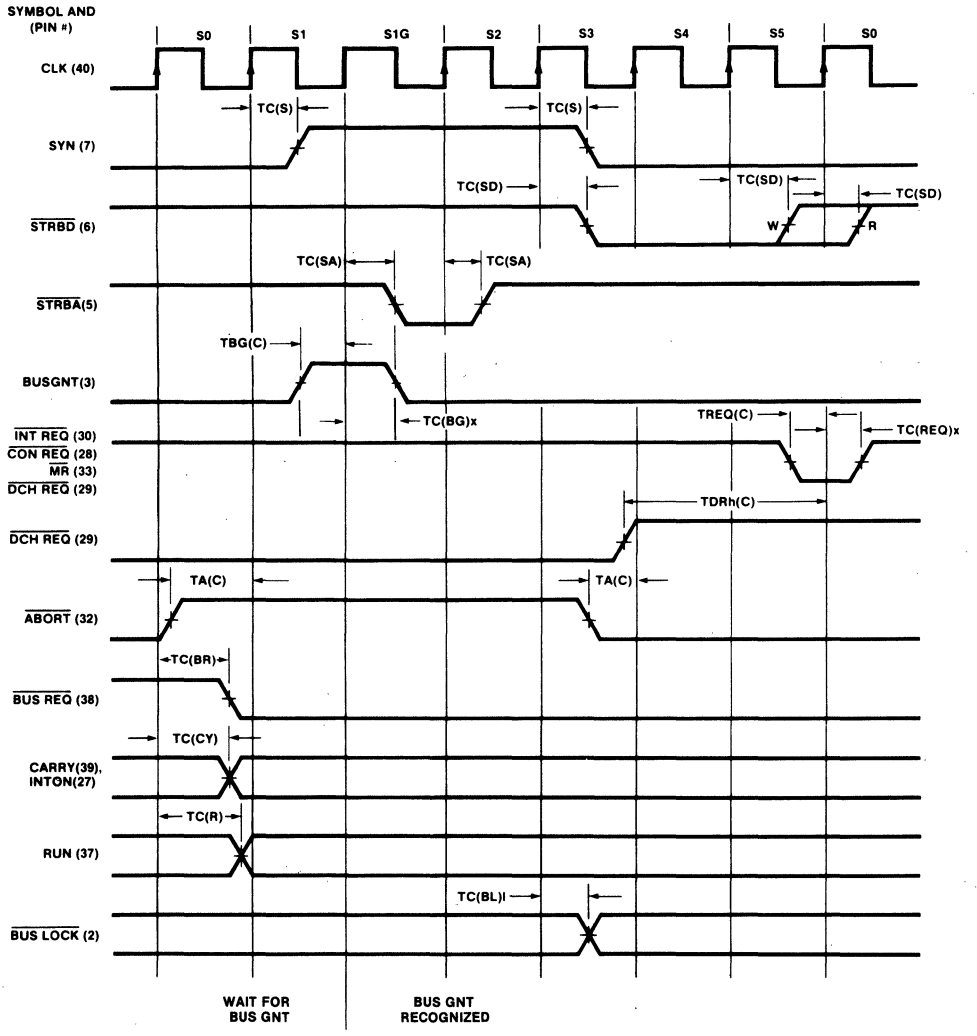
*See Timing Parameter Symbol Conventions at end of data sheet.

Fig. 14 Extended Memory Cycle Timing*



*See *Timing Parameter Symbol Conventions* at end of data sheet.

Figure 15. Bus and Status Control Timing*



*See *Timing Parameter Symbol Conventions* at end of data sheet.

**If this DCH REQ set-up time is missed, it is not recognized for another complete cycle.

Guaranteed Operating Ranges

Part Number	Supply Voltage (V_{CC})			Case Temperature
	Min	Typ	Max	
F9445DC	4.75 V	5.0 V	5.25 V	0 to +75°C
F9445DM	4.5 V	5.0 V	5.5 V	−55 to +125°C

DC Characteristics

(Over guaranteed operating ranges unless other wise noted.)
I_{INJ}(min) = 375 mA; I_{INJ}(max) = 425 mA

Symbol	Characteristic	Min	Typ	Max	Unit	Test Conditions
V_{IH}	Input HIGH Voltage	2.0			V	Guaranteed Input HIGH Voltage
V_{IL}	Input LOW Voltage			0.8	V	Guaranteed Input LOW Voltage
V_{CD}	Input Clamp Diode Voltage		−0.9	−1.5	V	V_{CC} = Min, I_{IN} = −18 mA, I_{INJ} = MIN
V_{OH}	Output HIGH Voltage; RUN, CARRY, INTON, SYN, STRBD, BUSREQ, STRBA, O_0, O_1, M	2.4	3.4		V	V_{CC} = Min, I_{OH} = −400 μA, I_{INJ} = MIN
V_{OH}	Output HIGH Voltage; $\overline{IB}_{(0\text{-}15)}$, W	2.4	3.4		V	V_{CC} = Min, I_{OH} = −1.0 mA, I_{INJ} = MIN
V_{OL}	Output LOW Voltage		0.25	0.5	V	V_{CC} = Min, I_{OL} = 8.0 mA, I_{INJ} = MIN
I_{IH}	Input HIGH Current; DCHREQ, INTREQ, CLK, MR, RDYA, RDYD, ABORT, CONREQ, BUSGNT		2.0	40	μA	V_{CC} = Max, V_{IN} = 2.7 V, I_{INJ} = MIN
I_{IH}	Input HIGH Current; $\overline{IB}_{(0\text{-}15)}$ (3-state)		5.0	100	μA	V_{CC} = Max, V_{IN} = 2.7 V, I_{INJ} − MIN
I_{IH}	Input HIGH Current; All inputs			1.0	mA	V_{CC} = Max, V_{IN} = 5.5 V, I_{INJ} = MIN
I_{IL}	Input LOW Current; All Inputs		−0.21	−0.4	mA	V_{CC} = Max, V_{IN} = 0.4 V, I_{INJ} = MIN
I_{OZH}	Output OFF State (High Impedance) Current $\overline{IB}_{0\text{-}15}$, W			100	μA	V_{CC} = Max, V_{OUT} = 2.4 V, I_{INJ} = MIN
I_{OZL}	Output OFF State (High Impedance) Current $\overline{IB}_{0\text{-}15}$		−210	−400	μA	V_{CC} = Max, V_{OUT} = 0.4 V, I_{INJ} = MIN
I_{OZL}	Output OFF State (High Impedance) Current; $\overline{W}$			−100	μA	V_{CC} = Max, V_{OUT} = 0.5 V, I_{INJ} = MIN
I_{OSH}	Output Short Circuit Current; All Outputs Except BUSLOCK	−15		−100	mA	V_{CC} = Max, V_{OUT} = 0.0 V, I_{INJ} = MIN
I_{LOH}	Output Leakage; BUSLOCK			1.0	mA	V_{CC} = Min, V_{OH} = 5.25 V, I_{INJ} = MIN
I_{CC}	Supply Current		200	350	mA	V_{CC} = Max, I_{INJ} = 300 MIN
V_{INJ}	Injector Voltage		1.3		V	I_{INJ} = 400 mA

*Not more than one output to be shorted at a time.

AC Characteristics

T_C = 0 to 75°C; V_{CC} = 4.75 to 5.25 V; I_{INJ} = 375 mA; C_L = 15 pF.
Input conditioning: Rise Time = 6 ns; Fall time = 6 ns; Amplitude = 0 to 3 V

Refer to *Symbol Conventions* at the end of this data sheet for explanation of the timing parameter symbols.

Symbol	Characteristic	Min	Typ	Max	Unit
TC(O)	Propagation delay, CLK to O_0, O_1, $\overline{M}$		60		ns
TC(S)	Propagation delay, CLK to SYN		30		ns
TC(W)	Propagation delay, CLK to $\overline{W}$		70		ns
TC(W)z	Propagation delay, CLK to $\overline{W}$ going 3-state		70		ns
TC(IBA)	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, address		60		ns
TC(IBA)z	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, address, going 3-state (read cycle)		35		ns
TC(SA)	Propagation delay, CLK to $\overline{STRBA}$		30		ns
TC(IBD)	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, data out		75		ns
TC(IBD)z	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, data out, going three-state		35		ns
TC(SD)	Propagation delay, CLK to $\overline{STRBD}$		25		ns
TRA(C) TC(RA)x	Setup time, RDYA to CLK Hold time, CLK to RDYA		3 10		ns ns
TRD(C) TC(RD)x	Setup time, RDYD to CLK Hold time, CLK to RDYD		2 10		ns ns
TIBD(C)	Setup time, $\overline{IB}_{(0\text{-}15)}$, data in, to CLK (read or fetch cycle)		75		ns
TC(IBD)x	Hold time, $\overline{IB}_{(0\text{-}15)}$, data in, after CLK (read or fetch cycle)		25		ns
TREQ(C)	Setup time, $\overline{INTREQ}$, $\overline{DCHREQ}$, $\overline{CONREQ}$, $\overline{MR}$ to CLK, all are the same timing relative to S5		15		ns
TC(REQ)x	Hold time, $\overline{INTREQ}$, $\overline{DCHREQ}$, $\overline{CONREQ}$, $\overline{MR}$ after CLK, all are the same timing		20		ns
TDRh(C)	Data channel ($\overline{DCHREQ}$) off setup time from CLK (to finish data-channel cycle)		100		ns
TA(C)	Setup time, $\overline{ABORT}$ to CLK		30		ns
TC(BL)1	Propagation delay, CLK to $\overline{BUSLOCK}$ going LOW		35		ns
TBG(C) TC(BG)x	Setup time, BUSGNT to CLK Hold time, CLK after BUSGNT		10 10		ns ns
TC(R)	Propagation delay, CLK to RUN		80		ns
TC(CY)	Propagation delay, CLK to CARRY		50		ns
TC(INT)	Propagation delay, CLK to INTON		50		ns
TC(BR)	Propagation delay, CLK to $\overline{BUSREQ}$		40		ns

6

Timing Parameter Symbol Conventions

The abbreviated symbols used for ac characteristic timing parameters in this data sheet are defined as follows:

The timing symbol convention is: TAb(C)d

The timing symbols all begin with the letter "T".

The second position, represented by "A", indicates the signal node beginning the interval.

The position "b" defines the direction of signal transition at the beginning node "A", if such definition is necessary; the new state of the signal may be: l = Low; h = High; z = 3-state; x = Don't care; v = Valid

The position "C", which always appears within parentheses, indicates the signal node ending the interval.

The position "d" is the same as "b" but refers to the state of the signal at the node indicated by the mnemonic in position "C".

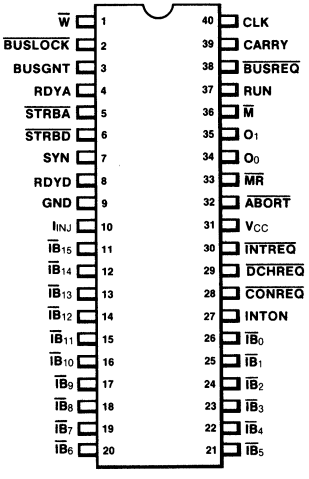
Ordering Information

ORDER CODE	SPEED	TEMPERATURE
F9445-24 DC	24 MHz	0°C to +75°C
F9445-24 DM	24 MHz	-55°C to +125°C
F9445-24 DMQB	24 MHz	-55°C to +125°C
F9445-20 DC	20 MHz	0°C to +75°C
F9445-20 DM	20 MHz	-55°C to +125°C
F9445-20 DMQB	20 MHz	-55°C to +125°C
F9445-16 DC	16 MHz	0°C to +75°C
F9445-16 DM	16 MHz	-55°C to +125°C
F9445-16 DMQB	16 MHz	-55°C to +125°C

For other temperature ranges, contact Fairchild Sales Office.

All packages are 40-pin ceramic DIPs

Connection Diagram
40-Pin DIP (Top View)



FAIRCHILD
A Schlumberger Company

F9446
Dynamic Memory Controller

Advance Product Information

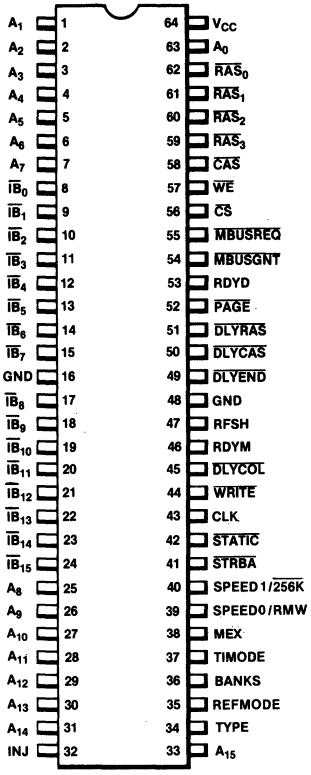
Microprocessor Product

Description

The Fairchild F9446 Dynamic Memory Controller (DMC) is designed to support a variety of memory configurations and provide an interface between 16K and 64K memory chips and the F9445 central processing unit (CPU). It provides a 16-bit memory address register (MAR), an address multiplexer for the row, column, and refresh addresses, a timing generator for the row and column strobe signals and the write enable signal (RAS-CAS-WE), mode arbitration, and page mode logic. It is implemented in I^3L® bipolar technology with low-power Schottky-compatible inputs and outputs.

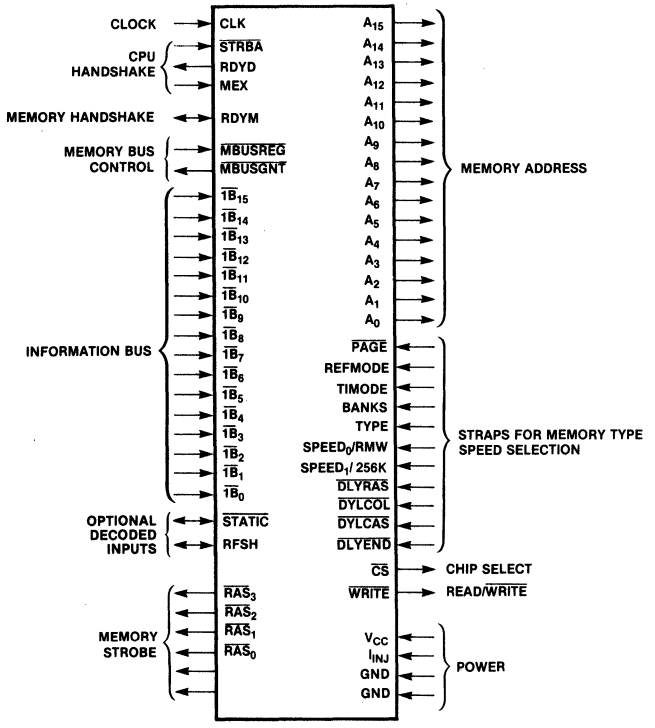
- **16-bit Memory Address Register**
- **Ability to Accommodate a Variety of Memory Speeds**
- **16K or 64K DRAMs**
- **Automatic Page Mode**
- **Internal Refresh Address Counter**
- **Row/Column/Refresh Multiplexer**
- **Complete Memory Timing Signals**
- **Three-state Outputs for Multiport Memories**
- **Internal Refresh Rate Timer**
- **Low-power Schottky-compatible I/0**
- **I^3L Bipolar Technology**
- **64-Pin DIP**
- **Operating Temperature Range of from −55°C to +125°C**

Connection Diagram



6

Signal Descriptions



The F9446 incorporates an address multiplexer and memory timing generator for use with both static and dynamic memories. The multiplexer selects between row and column segments of the internal 16-bit memory address register (MAR) or an internal refresh address counter (RAC). The upper two address bits provide bank information, while the lower seven, eight, or nine address bits are multiplexed to provide row, column, and refresh address. Assertion of the static line supresses row/column multiplexing of the memory address register outputs, which then provides the full 16-bit address.

The memory timing sequence is initiated by the memory execute signal; it may be inhibited or aborted by removal of the chip select signal. Once started, the memory timing sequence is automatic. A choice of four speed grades accommodates a variety of memory access times, and external controls may be used to further modify the timing.

The four individual RAS lines accommodate from one to four banks of memory chips, with automatic satisfaction of precharge requirements for memory access and refresh, in page mode or not, with distributed refresh or bulk refresh.

Refreshes are initiated to satisfy a rate of 128 per 2 ms per RAS bank. For short intervals, they may automatically be deferred until nonmemory CPU cycles; this makes them semi-transparent.

A memory bus request and memory bus grant are provided to govern 3-state control of memory interface signals for multiport or DMA purposes.

FAIRCHILD
A Schlumberger Company

F9447
I/O Bus Controller

Advance Product Information

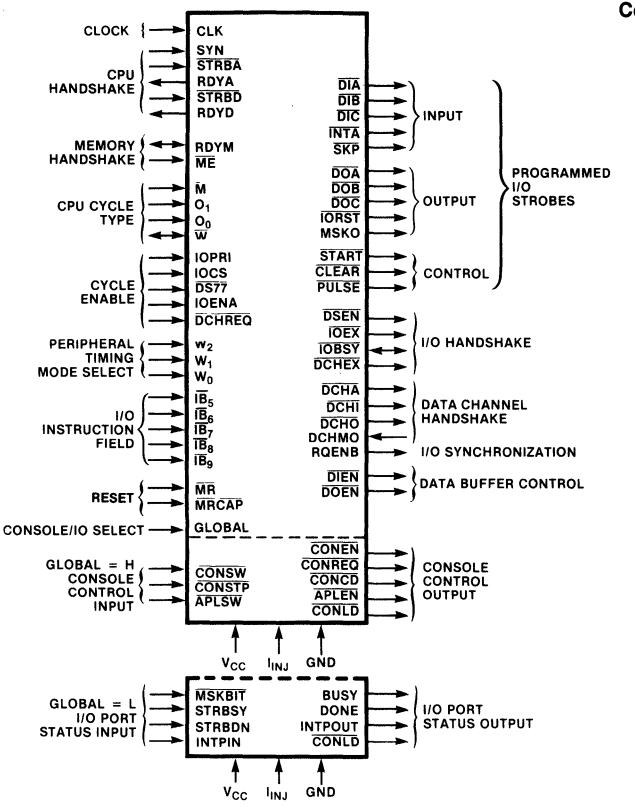
Microprocessor Product

Description

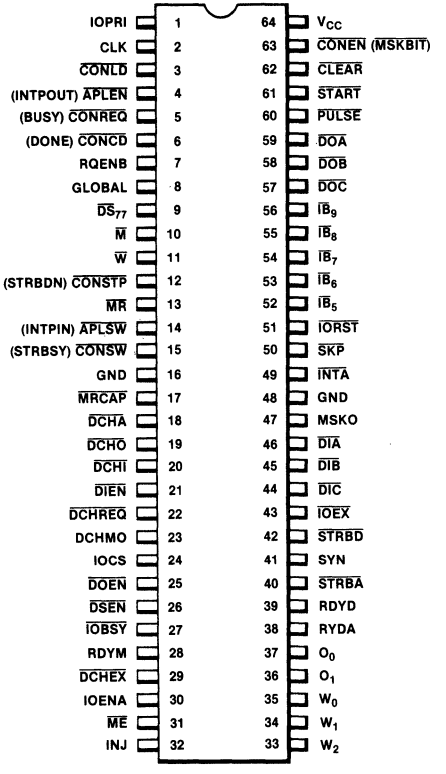
The F9447 I/O Bus Controller (IOC) is used with the F9445 I^3L® 16-Bit Bipolar Microprocessor to demultiplex the I/O instruction and data of the information bus (IB). It provides all the timing and decode signals required for programmed or data channel (DCH) input/output to peripheral device controllers. In the NOVA® -compatible mode, the F9447 provides all the timing and signals required by that I/O bus. For DCH transfers, address generation and handshake can be handled by the F9449 Multiple Data Channel Controller.

- **Interfaces Directly to the F9445**
- **Controls Standard and High-Speed NOVA-Compatible Data Channels**
- **NOVA-Compatible or F9445 I/O and Data Channel Timing**
- **Complete NOVA-Compatible I/O Bus Interface**
- **Automatic Program Load**
- **Power-Up Reset Delay**
- **Console Interface**
- **Local Busy/Done/Interrupt Logic**
- **Low-Power Schottky-Compatible I/O**
- **64-Pin DIP or Optional Chip Carrier Package**
- **I^3L Technology**
- **Operating Temperature Range of from −55°C to +125°C**

Signal Functions



Connection Diagram



6

® I^3L is a registered trademark of Fairchild Camera and Instrument Corp.

® NOVA is a trademark of Data General Corporation.

The F9447 I/O controller is a decoder and timing generator for programmed I/O instructions and data channel transfers. The timing sequence is selected via a three-bit control code, W_0-W_2. Additional logic is included to implement either the basic console interface or a busy/done/interrupt function. A hysteresis circuit for deriving a power-on-reset, from an external capacitor to ground, is also included. Figure 1 is a typical block diagram of the F9447 I/O controller. Figure 2 shows how the F9447 can be used with the F9445 system.

Signal Description

Table 1 describes the F9447 signals.

Figure 1 F9447 Block Diagram

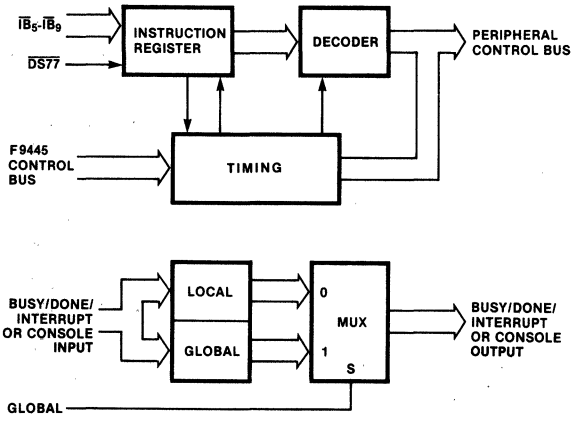


Figure 2 System Configuration

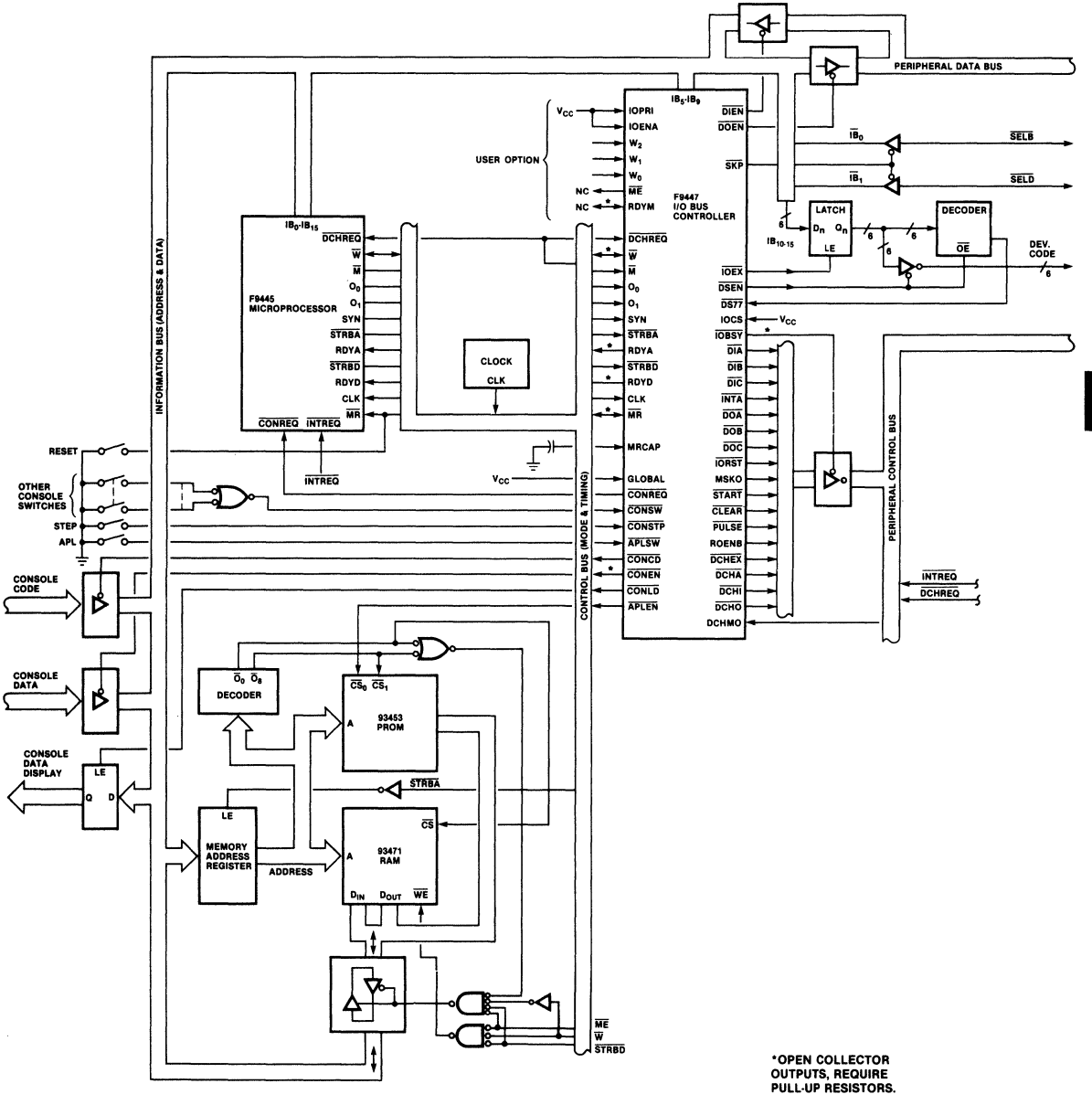


Table 1 F9447 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
CLK	2	Clock	A synchronizing input signal primarily for timing non-F9445 mode intervals through a state counter clock on alternate positive edges.
CPU Handshake			
SYN	41	Synchronize	An input signal from the CPU that maintains system timing.
$\overline{\text{STRBA}}$	40	Strobe Address	An active low input signal that indicates an address portion of the CPU cycle (and instruction during I/O execute).
RDYA	38	Ready Address	An active high open-collector output signal that allows the CPU to continue beyond the address portion of the cycle.
$\overline{\text{STRBD}}$	42	Strobe Data	An active low input signal that indicates the data portion of the CPU cycle.
RDYD	39	Ready Data	An active high open-collector output signal that allows the CPU to continue beyond the data portion of the cycle.
Memory Handshake			
RDYM	28	Ready Memory	A bidirectional, open collector signal; an active high input during a data channel output cycle indicates that the memory has fetched the data. An active high output during a data channel input cycle indicates to memory that the input data is valid and a write may proceed.
ME	31	Memory Enable	An active low output that modifies the memory controller to respond for CPU memory accesses or for data channel cycles ($ME = M + O_1 \cdot O_0$).
CPU Cycle Type			
$\overline{M}$	10	Memory	An active low input from the CPU that indicates a memory type of cycle.
O_1 O_0	36 37	O Lines	A pair of input signals from the CPU to indicate the type of cycle.
$\overline{W}$	11	Write	An active low input signal from the F9445 that indicates a cycle during which data is to be written to a memory or I/O device.

Mnemonic	Pin No.	Name	Description
Cycle Enable			
IOPRI	1	I/O Priority	An active high input that enables the F9447 to begin an I/O or data channel cycle.
IOCS	24	I/O Chip Select	An active high input enables strobes for busy and done, as well as I/O control decodes.
$\overline{DS}_{77}$	9	Device Select 77	An active low input signal that indicates a decode of $\overline{IB}_{10}$-$\overline{IB}_{15}$ all active, a device code 77 (CPU class) I/O instruction.
IOENA	30	I/O Enable	An active high input that, when low, inhibits response by the F9447 to any F9445-programmed I/O cycle. Used with multiple F9447s to allow disables.
$\overline{DCHREQ}$	22	Data Channel Request	An active low input that indicates there is a data channel request.
Peripheral Timing Mode Select			
W_2 W_1 W_0	33 34 35	Timing Options	A three-bit input code; the decodes provide a selection of timing for I/O and data channel cycles.
I/O Instruction Field			
$\overline{IB}_5$-$\overline{IB}_9$	52-56	Information Bus	Active low input signals from the F9445 containing I/O instruction bits 5 through 9 during address phase of I/O execute cycles.
Reset			
$\overline{MR}$	13	Master System Reset	An open-collector bidirectional pin that, when pulled low, initializes the F9447 and activates $\overline{IORST}$. Also an output generated by a low level on $\overline{MRCAP}$.
$\overline{MRCAP}$	17	Capacitive Reset	An active low input with an internal resistive pullup of 10K ohms to V_{CC}.

6

* The F9447 does not modify the F9445 timing for the following control instructions: READS, ION, APL, HALT.

Table 1 F9447 Signal Descriptions (Cont'd.)

Mnemonic	Pin No.	Name	Description
Console I/O Select			
GLOBAL	8	Global (Local)	An input that selects one of two uses of the global and local modes.
Programmed I/O Strobes			
$\overline{DIA}$	46	Data-In-A	Active low timing strobe outputs that indicate execution of data input instructions.
$\overline{DIB}$	45	Data-In-B	
$\overline{DIC}$	44	Data-In-C	
$\overline{INTA}$	49	Interrupt Acknowledge	
$\overline{SKP}$	50	Skip	
$\overline{DOA}$	59	Data-Out-A	Timing strobe outputs (all active low except active high MSKO) that indicate execution of data output instructions.
$\overline{DOB}$	58	Data-Out-B	
$\overline{DOC}$	57	Data-Out-C	
$\overline{IORST}$	51	I/O Reset	An active low output that indicates either a decode of the execution of the $\overline{IORST}$ instruction or a system reset caused by $\overline{MR}$ or $\overline{MRCAP}$ active.
MSKO	47	Mask Out	
$\overline{START}$	61	Start	Active-low control outputs that indicate decode and time of start, clear, and pulse control functions during programmed I/O execution.
$\overline{CLEAR}$	62	Clear	
$\overline{PULSE}$	60	Pulse	
I/O Handshake			
$\overline{DSEN}$	26	Device Select Enable	An active low output that enables device select.
$\overline{IOEX}$	43	I/O Execute	An active low output indicating that the F9447 is involved in the execution of a programmed I/O cycle.
$\overline{IOBSY}$	27	I/O Busy	An active low input indicating that another source is using the I/O bus. An active low output is provided when the F9447 is about to or is executing an I/O cycle.
$\overline{DCHEX}$	29	Data Channel Execute	An active low output indicating that a data channel transfer is in progress.
Data-Channel Handshake			
$\overline{DCHA}$	18	Data Channel Address	An active low timing strobe output that defines address transfer time of a data channel cycle.
$\overline{DCHI}$	20	Data Channel In	An active low timing strobe output that defines data transfer in (write to memory) of a data channel cycle.

Mnemonic	Pin No.	Name	Description
$\overline{\text{DCHO}}$	19	Data Channel Out	An active low timing strobe that is output during data transfer out (write to peripheral) of a data channel.
DCHMO	23	Data Channel Mode Out	An active high input results in data channel output cycles.
I/O Synchronization			
RQENB	7	Request Enable	A timing output that synchronizes interrupt and data-channel priorities.
Data Buffer Control			
$\overline{\text{DIEN}}$	21	Data In Enable	An active low output that enables peripheral data onto the F9445 information bus during programmed I/O or data-channel input cycles.
$\overline{\text{DOEN}}$	25	Data Out Enable	An active low output that enables information bus data onto the peripheral data bus during programmed I/O or data-channel output cycles.
Console Control Input			
$\overline{\text{CONSW}}$	15	Console Switch	An active low input with internal 2.4K-ohm pullup resistor to V_{CC} and digital delay of approximately 3 ms to eliminate contact bounce.
$\overline{\text{CONSTP}}$	12	Console Step	An active low input with characteristics of CONSW that initiates a console request lasting for two console code cycles of the F9445.
$\overline{\text{APLSW}}$	14	Auto Program Load Switch	An active low input with characteristics of CONSW and CONSTP that initiates a console request cycle with APL enable active.
Console Control Output			
$\overline{\text{CONEN}}$	63	Console Enable	An active low output to enable a console to provide information to the IB of the F9445 during a read or write operation.
$\overline{\text{CONREQ}}$	5	Console Request	An active low output to the F9445 to request console service.
$\overline{\text{CONCD}}$	6	Console Code	An active low output to enable the console code onto the IB of the F9445 in response to a console code cycle of the F9445.
$\overline{\text{APLEN}}$	4	Auto Program Load Enable	An active low output initiated by an APLSW input or the execution of (DOA ac, CPU) instruction and terminated by the execution of a (DOAP ac, CPU) instruction or system reset.
$\overline{\text{CONLD}}$	3	Console Load	An active low output to enable the console to latch data from the IB of the F9445.

Table 1 F9447 Signal Descriptions, Cont'd.

Mnemonic	Pin No.	Name	Description
I/O Port Status Input			
$\overline{\text{MSKBIT}}$	63	Mask Bit	The local logic contains an interrupt disable flag loaded from a select bit of the F9445 information bus during the execution of a mask out instruction. The MSKBIT signal is driven by that selected bit of the IB; a low level at the beginning of MSKO execution sets the interrupt disable flag.
STRBSY	15	Strobe Busy	A negative-going input edge that strobes the BUSY flag to the clear state.
STRBDN	12	Strobe Done	A negative-going input edge that strobes the DONE flag to the true state.
INTPIN	14	Interrupt Priority In	An active high input that determines which peripheral device may interrupt.
I/O Port Status Output			
BUSY	5	Busy Flag	An active high output of a flip-flop is set by the execution of an I/O-Start cycle with I/O Chip Select (IOCS) high; and cleared by a similar I/O-Clear cycle, by a negative transition on STRBSY, by execution of an IORST instruction, or by a low level on MR.
DONE	6	Done Flag	An active high output of a flip-flop is set by a negative transition on STRBDN; cleared by the execution of an I/O-Start or I/O-Clear while I/O Chip Select (IOCS) is high, by the execution of an IORST instruction, or by a low level on $\overline{\text{MR}}$.
INTPOUT	4	Interrupt Priority Out	An active high output that determines which peripheral device may interrupt.
$\overline{\text{CONLD}}$	3	Console Load	An active low output that enables the console to latch data from the F9445 information bus.
Power			
V_{CC}	64	Power Supply	Nominal +5 V DC.
I_{INJ}	32	Injection Current	Constant current (80 mA) obtained by using a dropping resistor from V_{CC} (nominal V_{INJ} = 1.3 V). Use of bypass capacitor to GND is desirable.
GND	16, 48	Ground	Common power and signal return.

FAIRCHILD
A Schlumberger Company

F9448
Programmable Multiport Interface

Advance Product Information

Microprocessor Product

Description

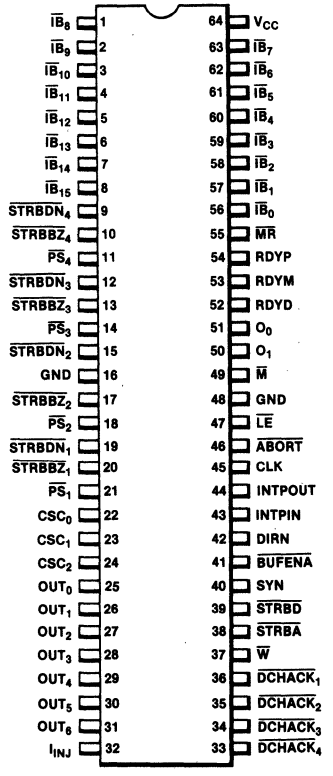
The F9448 Programmable Multiport Interface (PMI) is 64 pin bipolar I^3L device that facilitates the interface between an F9445 16-bit bipolar microprocessor and many industry-standard input/output (I/O) devices. It decodes I/O instructions and memory addresses from the central processing unit (CPU) to communicate with devices tied to its four external ports. When used with the F9449 Multiple Data Channel Controller, it handles peripheral selection and timing during data channel cycles. Some of the features of the F9448 are:

- **Four Independent I/O Ports**
- **Memory-Mapped and Programmed I/O**
- **Interface with Serial, Parallel, DMA, and Other Special-Purpose Devices**
- **Programmable Peripheral Timing**
- **Compatibility with Many Industry-Standard Interfaces**
- **Ability to Implement F9445-Programmed I/O Flags**
- **Interrupt Arbitration and Response Handling**
- **Fabricated in I^3L® Bipolar VLSI Technology**
- **Operating Temperature Range −55° to +125°C**
- **64-Pin Package**
- **Low-Power Schottky Compatible I/O**

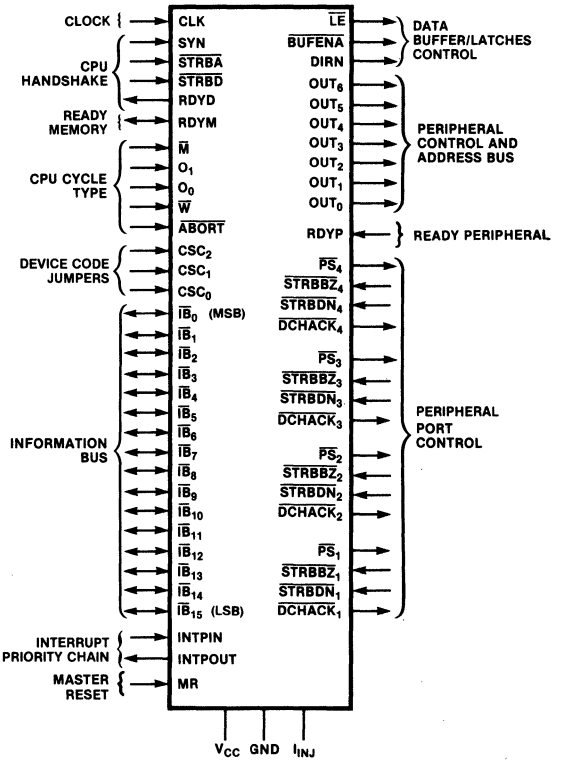
The F9448 PMI ties the F9445 CPU to many industry-standard microprocessor interfaces. It easily links I/O devices designed for the F6800 or 8080 buses and those directly suitable for the F9445 I/O bus. The system configuration in figure 1 shows how the F9448 can be used to interface F6800, 8086 family I/O devices or the F3870 to the F9445 system.

® I^3L is a registered trademark of Fairchild Camera and Instrument Corp.

Connection Diagram

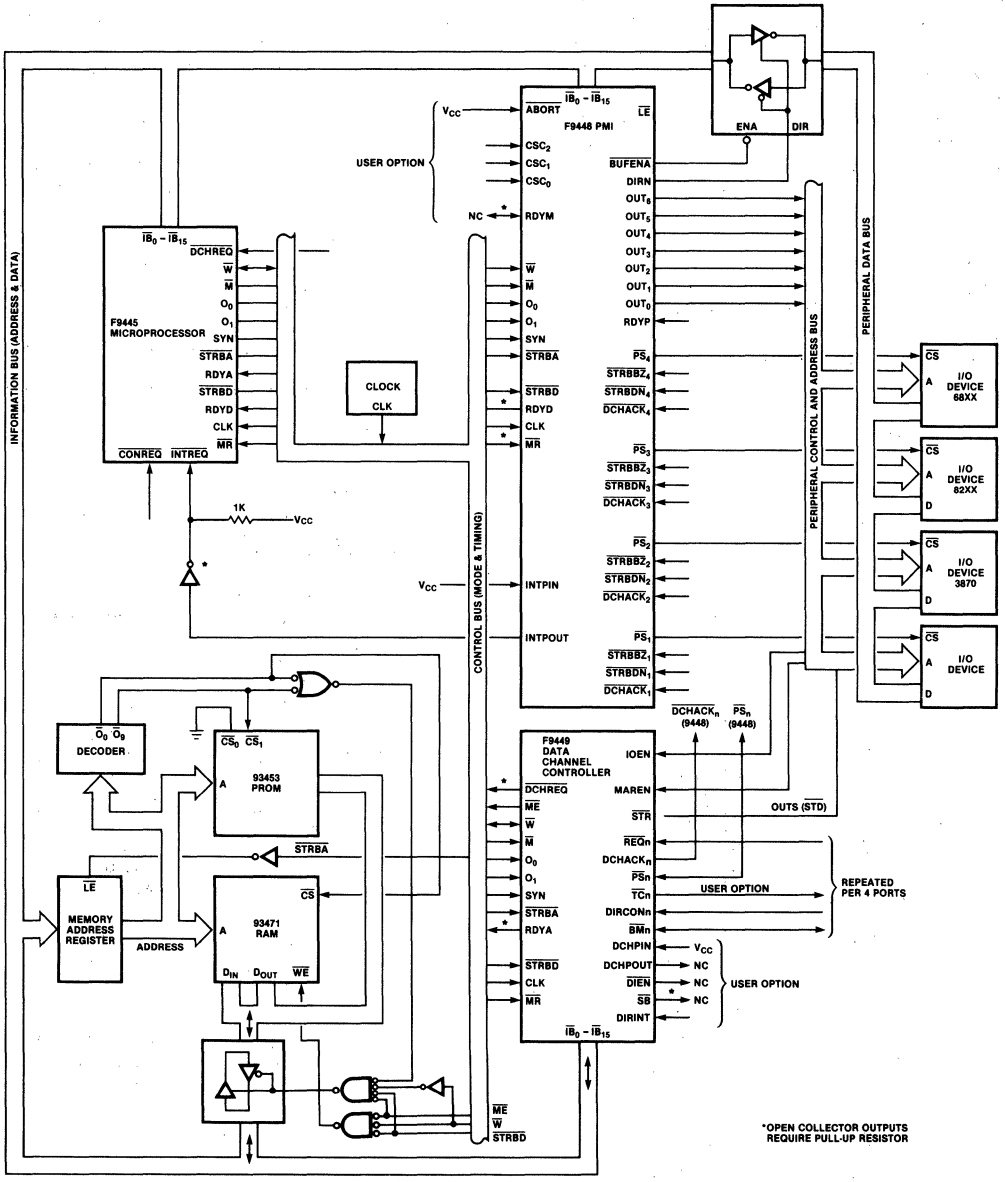


Signal Functions



6

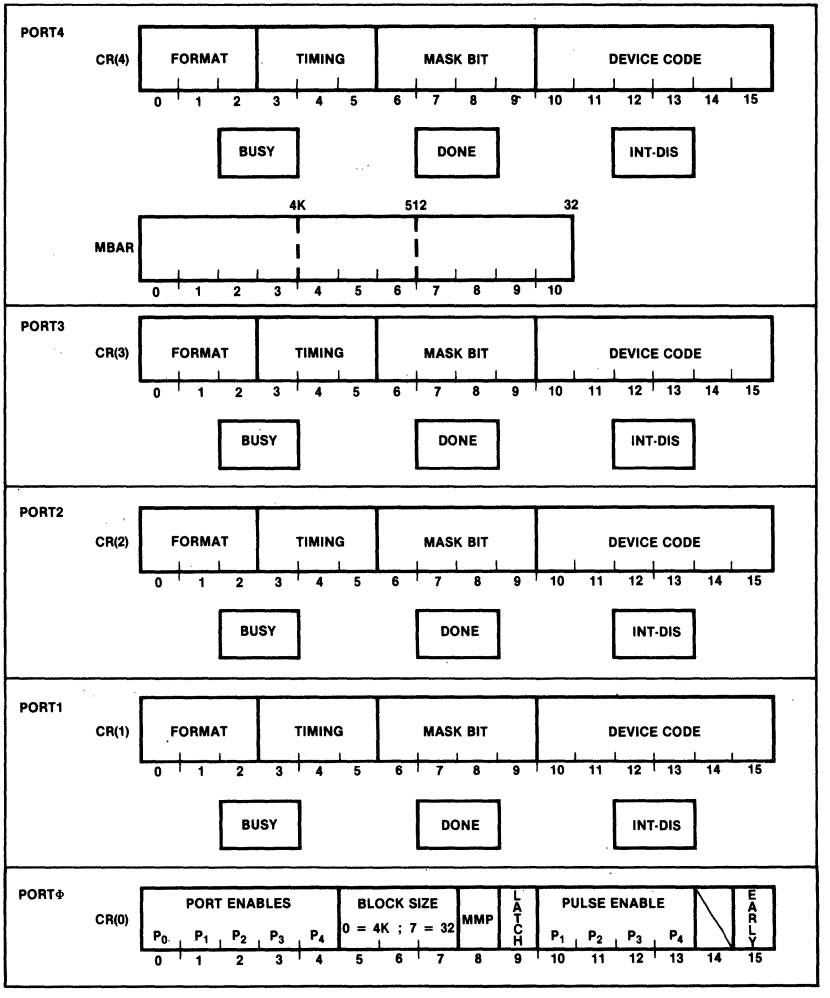
Figure 1 System Configuration



Registers

Each port has a 16-bit configuration register and busy, done, and interrupt-disable flags. Port 4 also has an 11-bit memory-base address register used in conjuction with memory-mapped I/O. Figure 2 shows the F9448 program-accessible registers.

Figure 2 F9448 Software Model



I/O Ports

The F9448 performs selecting and handshaking with connected peripheral devices. It has five bidirectional ports numbered 0 through 4. Port 0 is used as a bootstrap port by the F9445 to read and write, address and configuration registers inside the F9448; ports 1 through 4 are used to communiate with external peripheral devices. Ports 0 through 3 respond only to programmed I/O instructions; port 4 responds to either I/O instructions or memory cycles. This latter feature enables a block of up to 4096 memory addresses to be used for memory-mapped I/O.

Transactions between the F9448 and peripheral devices are organized using several select, address, and timing signals. Out 0-6 signals are shared by all ports, while each set of Peripheral Port Controls is associated with a specific port.

Signal Descriptions

Table 1 describes the signals for the F9448.

Table 1 F9448 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
CLK	45	Clock	Input signal from the positive-edge-triggered master clock from which all F9448 timing is generated.
CPU Handshake			
SYN	40	Synchronize	An input signal from the F9445 for synchronizing the F9445 with external devices. Active during every CPU cycle.
$\overline{\text{STRBA}}$	38	Strobe Address	An input signal generated by the F9445 during external bus cycles.
$\overline{\text{STRBD}}$	39	Strobe Data	An input signal generated by the F9445 during data transfer time and used by F9448 to organize transfers.
RDYD	52	Data Ready	An active high open collector output signal synchronizing F9448 with F9445 during data transfers. A low level stalls the F9445 until the peripheral is ready.
Ready Memory			
RDYM	53	Memory Ready	Handshake between the memory controller and F9448 during data-channel cycles. Input to F9448 during data-channel read from memory; output from F9448 during data-channel write to memory.
CPU Cycle Type			
$\overline{\text{M}}$	49	Memory	Input status lines from the F9445 indicating the type of bus cycle.
O_1	50	O-Line	
O_0	51	O-Line	
$\overline{\text{W}}$	37	Write	An input signal indicating the direction of data transfer on the IB.
$\overline{\text{ABORT}}$	46	Abort	A low input signal from the Memory Management and Protection Unit that prevents the F9448 from starting another cycle but allows completion of the current cycle.

Table 1 F9448 Signal Descriptions (Cont'd.)

Mnemonic	Pin No.	Name	Description
Device Code Jumpers			
CSC_0-CSC_2	22-24	Chip Select Code	Input signals tied to V_{CC}, GND, SYN, or SYN to define a 6-bit device code, DS_{10} through DS_{15} for port 0 of F9448.
Information Bus			
$\overline{IB}_0$-$\overline{IB}_{15}$	56-63, 1-8	Information Bus	A 16-bit, three-state address and data bus for transmitting information between the F9445 and external devices.
Interrupt Priority Chain			
INTPIN	43	Interrupt Priority Input	An input for determining which peripheral device will respond to an INTA instruction.
INTPOUT	44	Interrupt Priority Output	A priority-signal output from the F9448 to lower-priority devices for determining which peripheral device may interrupt.
Master Reset			
$\overline{MR}$	55	Master Reset	An input signal that initializes the F9448 by clearing all F9448 user-accessible registers to 0 and clearing all busy, done, and interrupt-disable flags.
Data Buffer/ Latches Control			
$\overline{LE}$	47	Latch Enable	An output signal that may be used to load the data from the F9445 on the IB into peripheral data bus latches.
$\overline{BUFENA}$	41	Buffer Enable	An active low output during memory, I/O, or data channel cycles to enable IB transceivers or latches if data is to be transferred between the IB and a device controlled by the F9448.
DIRN	42	Direction	An output signal controlling the direction of any bus transceivers on the IB bus between the F9445 and the F9448 or of any data latches/transceivers between the IB bus and a peripheral data bus.
Peripheral Control And Address Bus			
OUT_0-OUT_6	25-31	Outputs	A 7-bit peripheral output control bus that is to be shared by all peripheral devices controlled by F9448.
Ready Peripheral			
RDYP	54	Ready Peripheral	Open-collector handshake input signal from peripherals to the F9448.

Table 1 F9448 Signal Descriptions (Cont'd.)

Mnemonic	Pin No.	Name	Description
Peripheral Port Control			
$\overline{PS_1}$-$\overline{PS_4}$	21, 18, 14, 11	Port Select	Outputs for selecting the devices being controlled by ports 1 through 4 of the F9448.
$\overline{STRBBZ_1}$-$\overline{STRBBZ_4}$	20, 17, 13, 10	Strobe Busy	A low-to-high transition on the $\overline{STRBBZ}$ input signal clears the associated port's busy flag.
$\overline{STRBDN_1}$-$\overline{STRBDN_4}$	19, 15, 12, 9	Strobe Done	A low-to-high transition on the $\overline{STRBDN}$ input sets the associated port's done flag.
$\overline{DCHACK_1}$-$\overline{DCHACK_4}$	33-36	Data Channel Acknowledge	Active low select inputs from the F9449 data channel controller.
Power			
V_{CC}	64	Power Supply	Nominal +5 V DC.
GND	16, 48	Ground	Ground for both supply and signals.
I_{INJ}	32	Injection Current	A constant current obtainable by use of a dropping resistor from V_{CC} ($V_{INJ} \approx 13$ V) supply. Use of a bypass capacitor to GND is desirable.

FAIRCHILD
A Schlumberger Company

F9449

Multiple Data Channel Controller

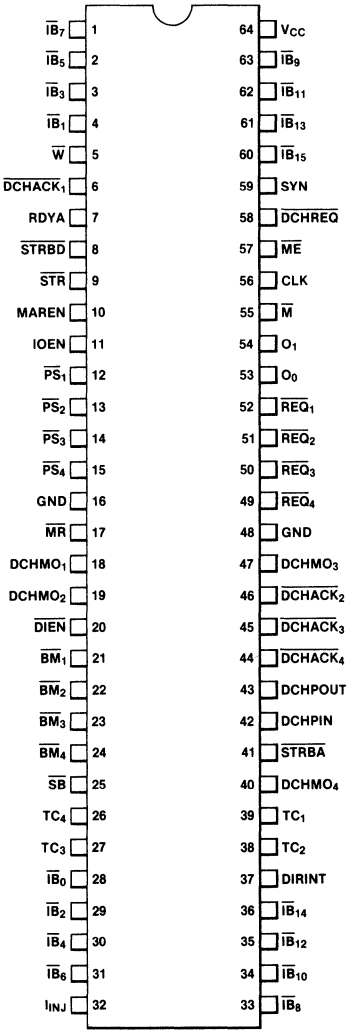
Microprocessor Product

Description

The F9449 Multiple Data Channel Controller is a 4-port controller that is used with the Fairchild F9445 16-Bit Bipolar Microprocessor, and either an F9447 I/O Bus Controller or an F9448 Programmable Multiport Interface, to control direct data transfer to and from memory by peripheral devices. It contains four pairs of program-controlled address and word count registers that are multiplexed to control four fully independent data channels (DCHs) through which data transfers can occur. Data channel transfers are similar to direct memory access (DMA) channel transfers, except that the F9445 architecture time-shares its information bus (IB).

- **Provides Control of Four Independent Channels**
- **Has Separate Word Count and Memory Address Registers for Each Channel**
- **Supports Byte- or Word-mode Operation on Each Channel**
- **Performs Internal Priority Arbitration**
- **Supports Memory-to-Memory Transfers**
- **Implemented in I³L® Technology, with Low-power Schottky TTL-compatible Input and Output**
- **Available in a 64-Pin Package.**
- **Operating Temperature Range of from −55°C to +125°C**

Connection Diagram



® I³L is a registered trademark of Fairchild Camera & Instrument Corp.

F9449 Signal Functions

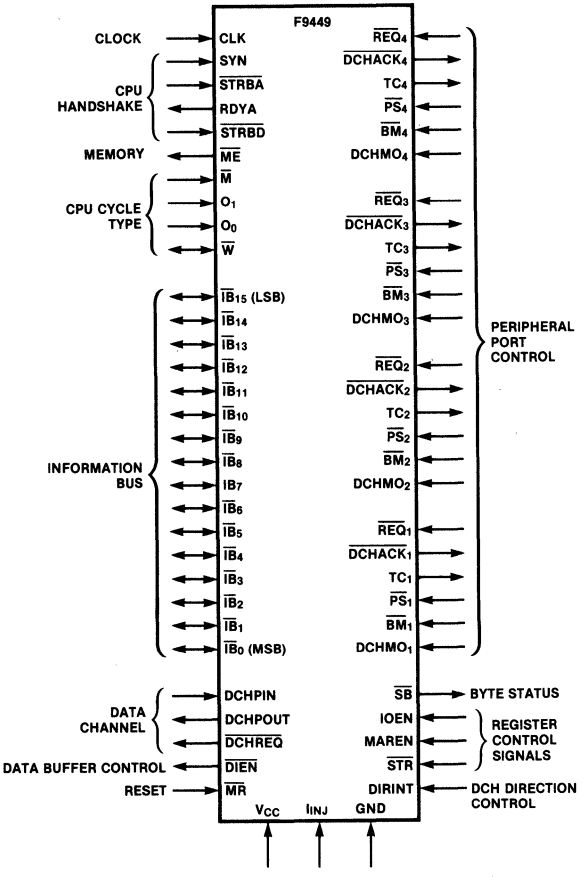
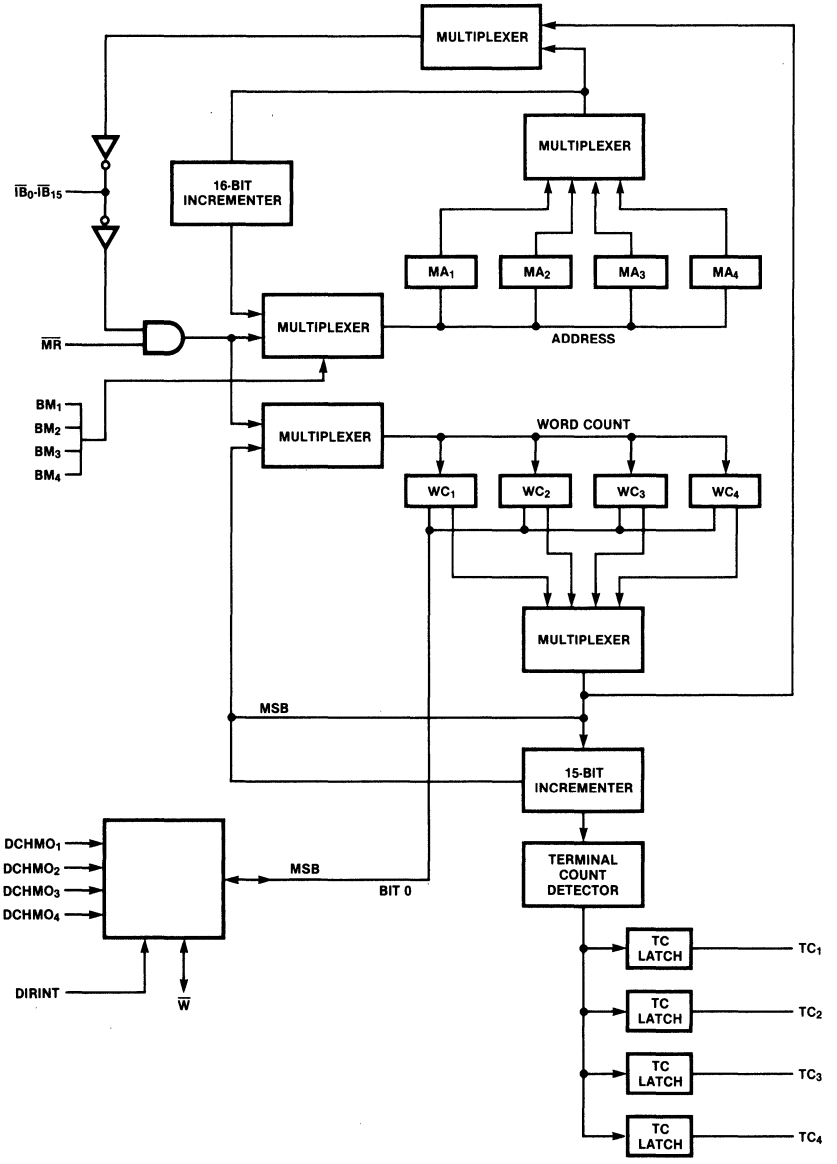


Fig. 1 F9449 Block Diagram



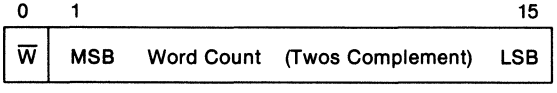
6

Register Operation

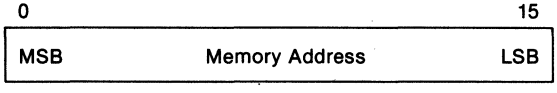
The eight F9449 registers (MA_{1-4}, WC_{1-4}), shown in figure 1, provide four fully independent data channels, numbered 1 to 4, each of which is capable of transferring up to 32K 8-bit bytes or 16-bit words, depending upon whether it is strapped for byte-mode or word-mode operation. Figure 2 illustrates the data formats of these registers.

A word count (WC) register associated with each channel contains the number of bytes or words to be transferred by that channel. The WC registers, which are loaded with the twos complement of the number of bytes or words, are automatically incremented after each DCH cycle (i.e., after every byte or word transfer), regardless of operating mode. When a WC register increments from all ones to zero, a terminal count (TC) signal for that port is set by the F9449. This is normally wired to the F9447 bus controller or F9448 multiport interface to generate an interrupt to the F9445. (Figures 3 and 4 illustrate system configurations using the F9447 and F9448, respectively.) The TC signal can optionally be used to terminate any further requests from that peripheral channel.

Figure 2 Register Data Formats



Format for word- or byte-count register load (MAREN = 0). Word count range is from -2^{15} to 1 (loaded with twos complement). $\overline{W}$ is an internal direction bit: 0 is from peripheral to memory; 1 is from memory to peripheral.



Format for memory address register load (MAREN = 1). Address range is from 0 to $2^{16}-1$.

A memory address (MA) register associated with each channel contains the address at which the next transfer is to occur; each MA register provides a 16-bit address space (0 to 65535). The MA registers are incremented after each transfer in word mode and after every second transfer in byte mode.

F9449 I/O Cycle

The F9449 registers are under software control. They are loaded with starting address and word count information through F9445 programmed output instructions, which are decoded by the F9447 I/O controller or F9448 multiport interface. The F9445 also generates the clock (CLK), synchronize (SYN), address strobe ($\overline{STRBA}$), and data strobe ($\overline{STRBD}$) bus timing signals. (Figure 5 illustrates the I/O cycle timing; refer to the "Timing Characteristics" section for a description of the cycle characteristics and specifications.)

The F9447 or F9448 selects the register to be loaded by generating the appropriate port select ($\overline{PS}$) signal, together with input/output enable (IOEN) and strobe ($\overline{STR}$) signals, as shown in table 1. The low-to-high transition of the STR signal during the write time loads the addressed port WC or MA register, selected by the memory address enable (MAREN) signal, with data from the information bus.

All eight registers are cleared when master reset ($\overline{MR}$) goes low to allow hardware implementation of auto load/bootstrap routines (i.e., to fill the memory beginning at address 0).

The F9445 can read the contents of any register by means of a programmed input instruction, which is decoded by the F9447 or F9448 in the same manner as the output instruction.

F9449 DCH Cycle

A peripheral device requests service by asserting its request ($\overline{REQ_n}$) line to the F9449, which then determines priority and generates a data channel request ($\overline{DCHREQ}$) signal to the F9445. After completing its current program instruction, the F9445 responds to the $\overline{DCHREQ}$ or data channel request performing a DCH cycle, which is a long bus cycle similar to an F9445 memory cycle, but with the information bus and the write ($\overline{W}$) line not driven.

The F9445 sets the bus control lines as follows: $\overline{M}$ high, O_1 low, and O_0 high (i.e., $\overline{M}$, O_1, O_0 to 101). It then generates the CLK, SYN, $\overline{STRBA}$ and $\overline{STRBD}$ bus timing signals. The high-to-low transition of $\overline{STRBA}$ latches the priority resolution logic, and starts the internal DCH sequence of the F9449. The F9449 asserts the appropriate data channel acknowledge ($\overline{DCHACK_n}$) line to signal the requesting peripheral and the F9447 or F9448 that a DCH cycle for it has begun. This may cause the peripheral to remove the REQ_n signal. (Figure 6 illustrates the DCH cycle timing; refer to the "Timing

Fig. 3 F9449/F9447 Configuration

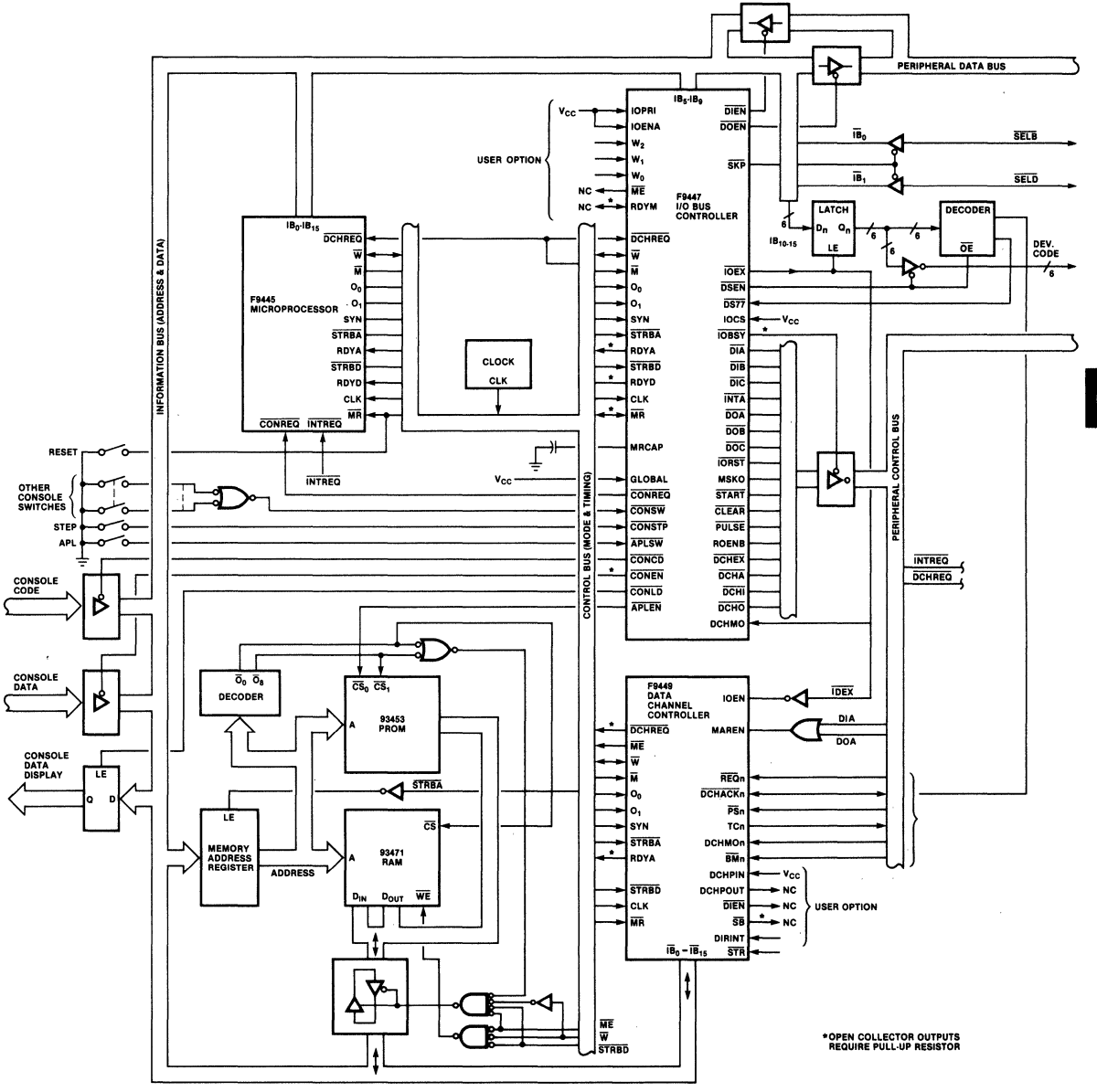


Fig. 4 F9449/F9448 Configuration

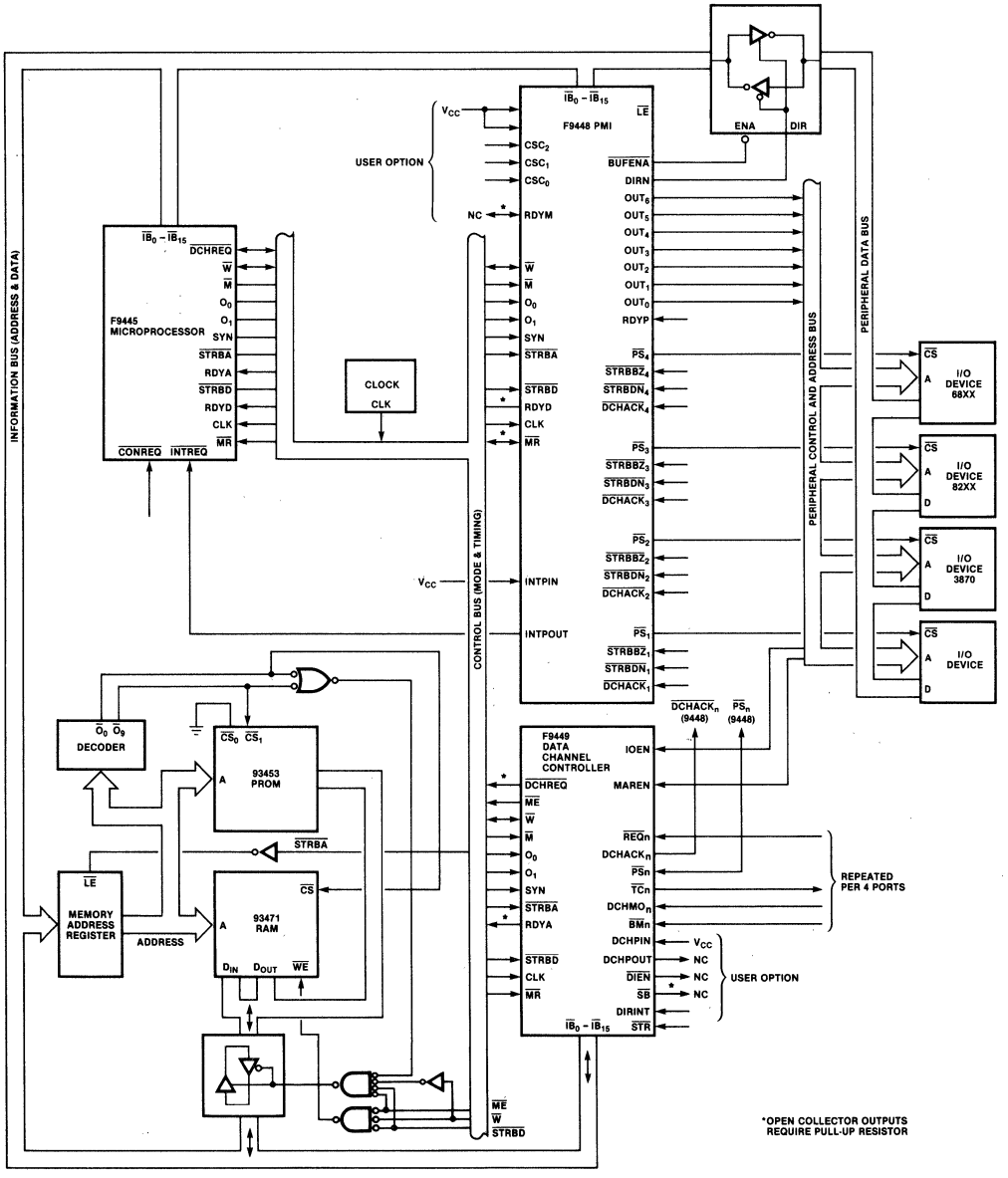


Table 1 F9449 I/O Control

Signal State								Operation Performed
IOEN	$\overline{W}$	MAREN	$\overline{PS_1}$	$\overline{PS_2}$	$\overline{PS_3}$	$\overline{PS_4}$	$\overline{STR}$	
0	X	X	X	X	X	X	X	No operation
X	X	X	1	1	1	1	1	No operation
1	0	0	*	*	*	*	U	Loads IB data into selected word count register
1	0	1	*	*	*	*	U	Loads IB data into selected memory address register
1	1	0	*	*	*	*	X	Loads selected word count register data onto IB
1	1	1	*	*	*	*	X	Loads selected memory address register data onto IB

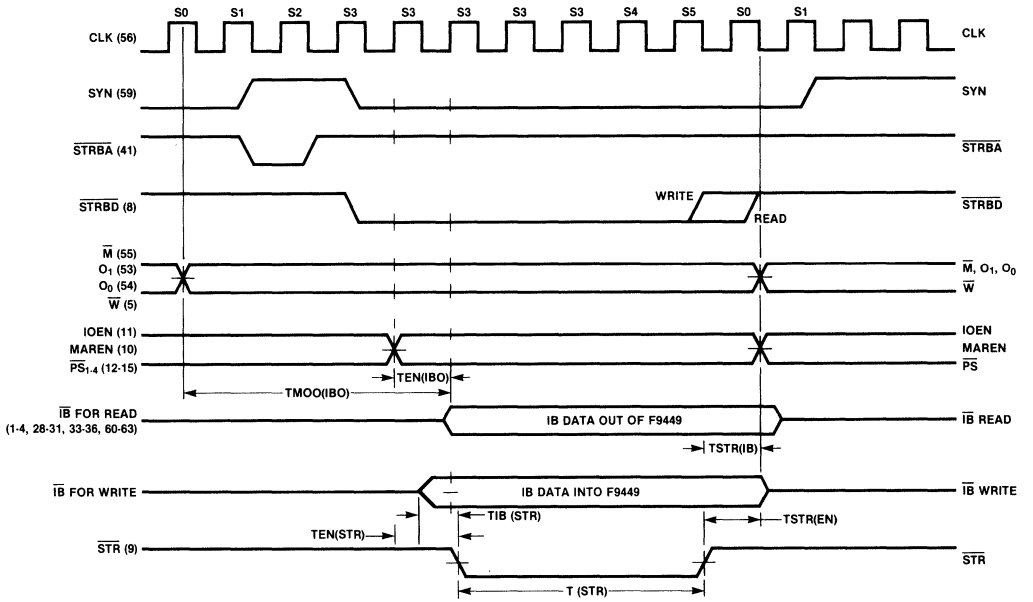
*One active-low input, selected by programmed I/O instruction device code.

Note
Multiple port selects result in unpredictable results.

0 = Low
1 = High
X = Don't care
U = Low-to-high transition

6

Fig. 5 Programmed I/O Timing



Characteristics" section for a description of the cycle characteristics and specifications.)

During the address phase of a DCH cycle, the F9449 determines which peripheral is to be served, places the contents of the appropriate MA register onto the information bus and drives the $\overline{W}$ line to the memory controller so that the memory performs either a read or a write cycle. If internal direction (DIRINT) is high, read/write selection can be programmed from the F9445 as the most significant bit of the WC register contents load. When this bit is at a logic 0 it causes a read of memory. If the DIRINT pin is low, the read/write selection is controlled by the peripheral through a data channel mode ($DCHMO_n$) signal. A high $DCHMO_n$ signal indicates a memory read (DCH out operation). If required, the data in enable ($\overline{DIEN}$) and second byte ($\overline{SB}$) lines are also asserted at this time.

The F9449 RDYA signal causes the microprocessor to generate three additional F9445 address strobe (S1G) states, allowing the address sufficient time to propagate from the F9449 to the memory controller.

The F9449 does not actually perform the data transfer between memory and peripheral. Instead, the end of the STRBA signal causes the F9449 to stop driving the address onto the information bus and allows the F9447 or F9448 to provide data control during the data phase of the DCH cycle. It enables a peripheral three-state input buffer, or strobes data out from the IB into the peripheral. The data phase of the DCH cycle can be extended as required by additional data (S3) states generated from the F9445 in response to the F9447 or F9448 data ready (RDYD) output being low.

Because it must communicate with the peripheral during F9445 programmed I/O cycles, the F9447 or F9448 normally accommodates the data timing peculiarities of the peripheral.

The end of the data strobe ($\overline{STRBD}$) causes the WC and MA registers to increment, a TC signal to be asserted (if the WC register has reached zero), and terminates the $\overline{W}$, $\overline{DCHACK}_n$, RDYA, and $\overline{SB}$ signals.

Priority Arbitration

The F9449 arbitrates DCH requests from multiple peripherals on a fixed-priority basis, with channel 1 having the highest priority and channel 4 the lowest. The priority arbitration scheme allows cascading of up to four F9449 controllers by interconnecting the data channel priority out (DCHPOUT) of a higher priority controller to data channel priority in (DCHPIN) of the next, thereby permitting the system to serve a total of 16 data channel peripherals.

Priority resolution occurs during every cycle, at the high-to-low transition of the SYN signal. In a multiple-F9449 system, all pending $\overline{REQ}_n$ inputs are latched at that time, and the DCHPIN/DCHPOUT signals ripple from device to device.

Priorities are reestablished during every cycle, including "short" F9445 cycles. Additional states are generated by the F9449 address ready (RDYA) signal to allow priority ripple when the F9445 responds to a DCH request from a "wait" cycle.

Signal Descriptions

The F9449 input and output signals are described in table 2.

Timing Characteristics

The timing characteristics of the F9449 are illustrated in figure 5 (Programmed I/O Timing) and figure 6 (Data Channel Cycle Timing).

The abbreviated symbol convention used for timing parameters in this data sheet is TAb(C)d, where:

- Timing symbols all begin with the letter "T".
- The mnemonic in the position represented by "A" indicates the signal node beginning the interval.
- The mnemonic in the position represented by "b" defines the direction of signal transition at the beginning node, if such definition is necessary; the new state of the signal may be low (l), high (h), 3-state (z), don't care (x), or valid (v).
- The mnemonic in the position represented by "C", which always appears in parentheses, indicates the signal node ending the interval.
- The mnemonic in the position represented by "d" is the same as "b", but refers to the state of the signal at the node indicated by the mnemonic in position "C".

Figure 6 Data Channel Cycle Timing

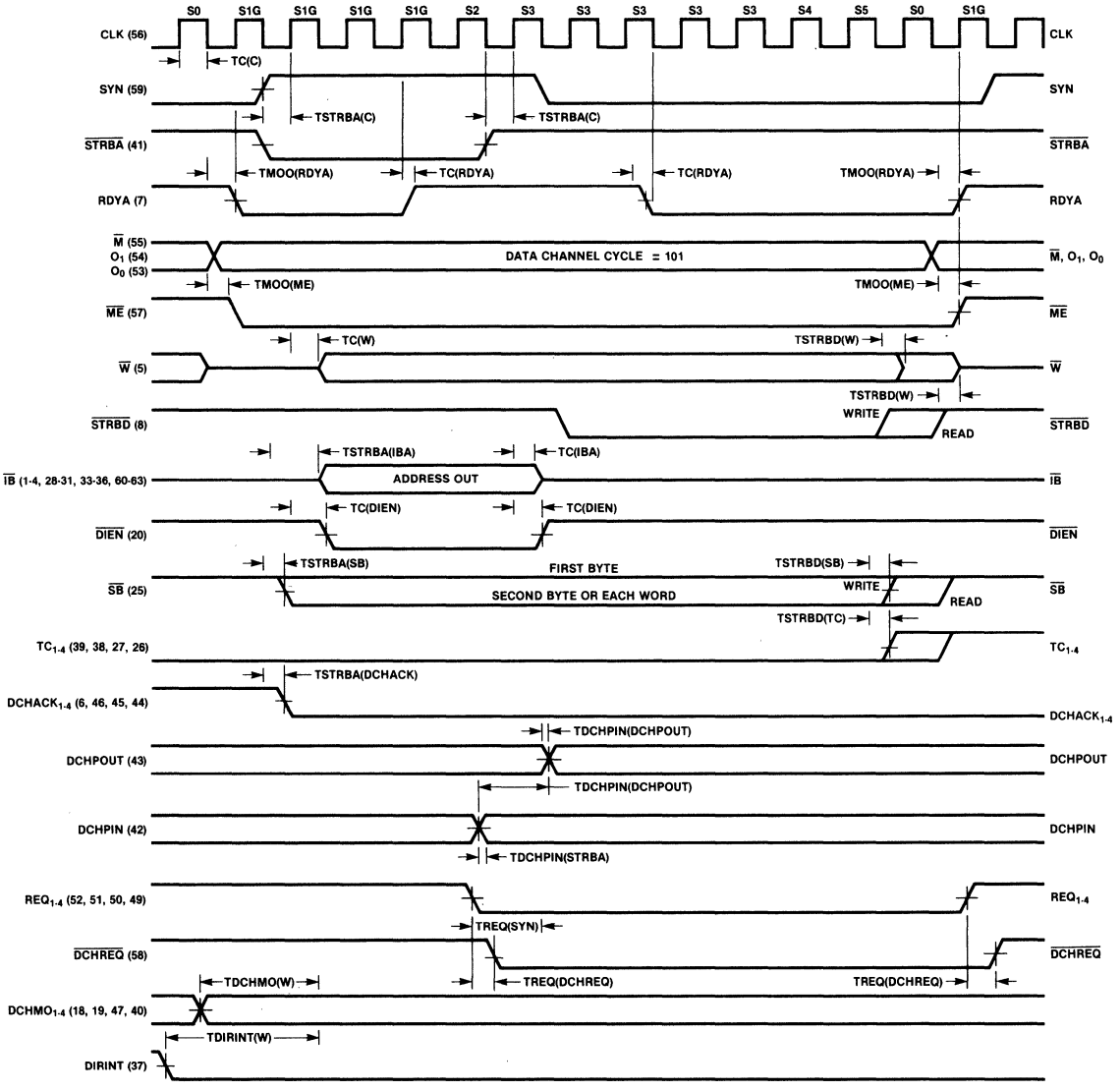


Table 2 F9449 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Clock			
CLK	56	Clock	An input signal from the F9445. The rising edge of the single-phase system clock causes action in the F9449. This line can also be single-stepped for debugging.
CPU Handshake			
SYN	59	Synchronize	An active-high input signal from the CPU that maintains system timing. The start of SYN indicates the start of a CPU cycle with valid $\overline{M}$, O_1, O_0 code.
$\overline{STRBA}$	41	Address Strobe	An active-low input signal from the F9445. In a DCH cycle, the low-to-high transition is used by the memory controller to strobe the memory address from the IB into the selected MA register. (This can be delayed indefinitely by RDYA.)
RDYA	7	Address Ready	An active-high open-collector output signal to the F9445. A low level prolongs the $\overline{STRBA}$ signal to allow time for the F9449 to perform priority resolution and propagate the memory address to the memory controller over the IB.
$\overline{STRBD}$	8	Data Strobe	An active-low input signal from the F9445. The low-to-high transition during a DCH cycle causes the selected WC and MA registers to increment and the $\overline{W}$, $\overline{DCHACK}_n$, RDYA and SB signals to terminate.
Memory			
$\overline{ME}$	57	Memory Enable	An active-low output signal to the memory controller. When low, it informs the memory controller that either the F9445 $\overline{M}$ is low or a DCH cycle is in progress.
CPU Cycle Type			
$\overline{M}$	55	Memory	An active-low input signal from the F9445 that serves as a status indicator. When it is low, the F9445 is performing a memory cycle.
O_1 O_0	54 53	Memory or I/O Function	Active-high "O" line input signals from the F9445, used with the $\overline{M}$ input to indicate the type of cycle the F9445 is performing. During a DCH cycle, $\overline{M}$, O_1, O_0 are set at 101.
$\overline{W}$	5	Write	An active-low input/output signal to and from the components in the system, normally driven by the F9445 to control the read and write operations. Placed in a high-impedance state by the F9445, during a DCH cycle, when it is driven by the F9449, it is low if the system is writing to memory and high if the system is reading from memory.

Table 2 F9449 Signal Descriptions (Cont.)

Mnemonic	Pin No.	Name	Description
Information Bus Signals			
$\overline{IB}_0$-$\overline{IB}_{15}$	1-4 28-31 33-36 60-63	Information Bus	A set of 16 input/output signals to and from the system. This active-low, bidirectional bus is used to load and examine the contents of the selected WC and MA registers. These signals are driven by the selected MA register during the $\overline{STRBA}$ state of an F9449 DCH cycle. The most significant bit is $\overline{IB}_0$; the least significant bit is $\overline{IB}_{15}$.
Data Channel			
DCHPIN	42	Data Channel Priority Input	An active-high input signal from a higher-priority F9449 that is used to extend priority resolution logic throughout a multiple-F9449 system. When this signal is low, it prevents the F9449 from being in a DCH cycle. The highest priority F9449 should have DCHPIN connected high.
DCHPOUT	43	Data Channel Priority Output	An active-high output signal to a lower-priority F9449 that is used to extend priority resolution logic throughout a multiple-F9449 system. When the signal is high, none of the four channels are requesting a DCH cycle and DCHPIN is high.
$\overline{DCHREQ}$	58	Data Channel Request	An active-low open-collector output used by the F9449 to request a DCH from the F9445. Multiple simultaneous requests will be sorted by priority resolution logic during a DCH cycle and will result in additional consecutive DCH cycles. A low level requests a data channel cycle.
Data Buffer Control			
$\overline{DIEN}$	20	Data in Enable	An active-low output signal that can be used to enable an optional bus transceiver placed between the F9449 and the IB. When low, the F9449 is putting out an address during a DCH cycle or data during an I/O read operation.
Reset			
$\overline{MR}$	17	Master Reset	An input signal that is active-low from a power-up, front-panel, or programmed initialization signal. It is used to load the WC and MA registers with zeros, set the internal direction control bit to zero, set the four TC signal lines high and clear the four $\overline{DCHACK}$ lines.
Peripheral Port Control			
$\overline{REQ}_1$-$\overline{REQ}_4$	52 51 50 49	Port Request	A set of four active-low input signals from the corresponding requesting peripherals. A low signal on a $\overline{REQ}_n$ line indicates that its associated peripheral wishes a DCH cycle. Priority resolution logic arbitrates multiple requests and generates a single acknowledge, $\overline{REQ}_1$ having the highest priority and $\overline{REQ}_4$ the lowest.

6

Table 2 F9449 Signal Descriptions (Cont.)

Mnemonic	Pin No.	Name	Description
$\overline{DCHACK}_1$- $\overline{DCHACK}_4$	6 46 45 44	Data Channel Acknowledge	A set of four active-low output signals to the requesting peripherals and to the F9447 or F9448. When low, it informs the appropriate peripheral that its requested DCH cycle is in progress. The $\overline{DCHACK}$ signal is used by the peripheral to clear the $\overline{REQ}_n$ line. It is also used by the F9447 or F9448 and by the peripheral to enable data buffers to and from the IB.
TC_1-TC_4	39 38 27 26	Terminal Count	A set of four active-high output signals to the associated peripherals, indicating completion of a DCH block. When a WC register is incremented to zero during the last phase of a DCH cycle, the corresponding TC line goes high. When the WC register is loaded with any value from the IB during an I/O write operation to the F9449, the corresponding TC line is cleared to low. All four TC signals are set high by a low level on MR.
$\overline{PS}_4$	12 13 14 15	Port Select	A set of four active-low input signals from a programmed I/O device. The IB bits are decoded by an F9448, which outputs a port select signal to the F9449. When low, the associated port is selected during an I/O read or write operation. No more than one PS line should be low at a time.
$\overline{BM}_1$-$\overline{BM}_4$	21 22 23 24	Byte Mode	A set of four active-low input lines that are used to establish operating modes. When strapped low, the corresponding channel is set for 8-bit byte-mode operation; when strapped high, the associated channel is set for 16-bit word-mode operation.
$DCHMO_1$- $DCHMO_4$	18 19 47 40	Data Channel Mode Out	A set of input signals from the requesting peripherals. When DIRINT is low, a $DCHMO_n$ low indicates that the corresponding peripheral is writing to memory during a DCH cycle (IN). When the $DCHMO_n$ signal is high, it indicates that the peripheral is reading from memory (OUT).
BYTE Status			
$\overline{SB}$	25	Second Byte	An active-low open-collector output signal to the memory controller. During $\overline{STRBD}$ timing, this signal is high during the first byte of a byte-mode DCH cycle and low during the second byte of a byte-mode cycle and during every word in a word-mode DCH cycle. It can be used to strobe either the left or right half of the memory array during a $\overline{STRBD}$ operation.
Register Control Signals			
IOEN	11	Input/Output Enable	An active-high input signal from the F9447 or F9448. It is used to enable the F9449 when the F9445 wishes to read from or write to a WC or MA register during an I/O cycle. When the signal is high, a programmed I/O operation is in progress.

Table 2 F9449 Signal Descriptions (Cont.)

Mnemonic	Pin No.	Name	Description
MAREN	10	Memory Address Register Enable	An active-high input signal from the F9447 or F9448. It is used to select the source/destination register for an F9445 programmed I/O operation. A high signal selects an MA register; a low signal selects a WC register.
$\overline{STR}$	9	Strobe	An input signal from the F9447 or F9448. The low-to-high transition causes the F9449 to load the IB data into the selected WC or MA register during an I/O write operation.
DCH Direction Control			
DIRINT	37	Internal Direction	Input line that is used to establish the control source of the $\overline{W}$ line. When DIRINT is high during a DCH cycle, the $\overline{W}$ line is controlled internally by IB_0, the most significant bit of the data word in the WC register. When DIRINT is low, the $\overline{W}$ line is controlled externally by the $DCHMO_n$ input from the corresponding requesting peripheral. It may be driven low by selected $\overline{DCHACK}_n$ outputs if some channels need internal control and others external control.
Power			
V_{CC}	64	Power Supply	Supply voltage (+5 Vdc).
I_{INJ}	32	Injection Current	A constant 250 mA current supply; may be derived by use of an external resistor to V_{CC}. (Nominal V_{INJ} = 1.2 V.)
GND	16, 48	Ground	Common power and signal return.

Table 3 DC Characteristics

Symbol	Characteristic	Min	Typ	Max	Unit	Test Conditions
V_{INJ}	Injector Voltage.		1.3		V	I_{INJ} = Max
V_{IH}	Input High Voltage.	2.0			V	Guaranteed Input High Voltage
V_{IL}	Input Low Voltage.			0.8	V	Guaranteed Input Low Voltage
V_{CD}	Input Clamp Diode Voltage.		-0.9	-1.5	V	V_{CC} = Min, I_{IN} = -18 mA, I_{INJ} = Min
V_{OH}	Output High Voltage.	2.4	3.2		V	V_{CC} = Min, I_{OH} = -400 μA, I_{INJ} = Min
V_{OL}	Output Low Voltage.		0.2	0.5	V	V_{CC} = Min, I_{OL} = 8.0 mA, I_{INJ} = Min
I_{IH}	Input High Current All Inputs.			1.0	mA	V_{CC} = Max, V_{IN} = 5.5 V, I_{INJ} = 300 mA
I_{IL}	Input Low Current.		-0.21	-0.4	mA	V_{CC} = Max, V_{IN} = 0.4 V, I_{INJ} = Min
I_{OZH}	Output Off (High-Impedance) State High Current IB_0-IB_{15}, W.			100	μA	V_{CC} = Max, V_{OUT} = 2.4 V, I_{INJ} = Min
I_{OZL}	Output Off (High-Impedance) State Low Current IB_0-IB_{15}, W.		-210	-500	μA	V_{CC} = Max, V_{OUT} = 0.4 V I_{INJ} = Min
I_{OSH}	Output Short Circuit Current.	-15		-100	mA	V_{CC} = Max, V_{OUT} = 0.0 V, I_{INJ} = Min
I_{LOH} O_{HH}	Output Leakage Current (Open Collector) RDYA, SB, DCHREQ.			1.0	mA	V_{CC} = Min, V_{OH} = 5.25 V, I_{INJ} = Min
I_{CC}	Supply Current.		125		mA	V_{CC} = Max, I_{INJ} = Min

*Not more than one output to be shorted at a time.

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this data sheet, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Storage Temperature	−65 °C, +150 °C
Ambient Temperature Under Bias	−55 °C, +125 °C
V_{CC} Pin Potential to Ground Pin	−0.5 V, +6.0 V
Input Voltage (dc)	−0.5 V, +5.5 V
Input Current (dc)	−20 mA, +5 mA
Output Voltage (Output HIGH)	−0.5 V, +5.5 V
Output Current (dc) (Output LOW)	+20 mA
Injector Current (I_{INJ})	+500 mA
Injector Voltage (V_{INJ})	−0.5 V, +2.0 V

Recommended Operating Ranges

Part Number	Supply Voltage (V_{CC})		
	Min	Typ	Max
F9449DC	4.75 V	5.0 V	5.25 V
F9449DM	4.5 V	5.0 V	5.5 V

Part Number	Injector Current (I_{INJ})		
	Min	Typ	Max
F9449DC	200 mA	250 mA	300 mA
F9449DM	200 mA	250 mA	300 mA

Ordering Information

Order Code	Temperature Range
F9449DC	0 °C to +75 °C
F9449DM	−55 °C to +125 °C

FAIRCHILD
A Schlumberger Company

F9450 (MIL-STD 1750A) 16-Bit Bipolar Microprocessor

Advance Product Information

Microprocessor Product

- **Single-Chip Microprocessor Fully Implements MIL-STD 1750A (Notice 1) ISA.**
- **High Performance Over Military Temperature Range: 700K IPS DAIS Mix with Floating Point; 0.2 μs ADD, 1.85 μs MULTIPLY**
- **Real-Time Processing, Two Programmable Timers, 16 Levels of Vectored Interrupt**
- **32- and 48-Bit Floating Point Arithmetic on Chip**
- **Bipolar VLSI I^3L® -II - 1 x 10^5 Radiation**
- **Multiprocessor Capabilities**
- **Single and Double Precision Arithmetic**
- **Direct Address of Up to 64K Words, Expandable to 1M Words**
- **16 General-Purpose Registers**
- **Static Operation with Single Clock (0-20 MHz)**
- **Low-Power Schottky Inputs and Outputs**
- **Single 5 V Supply; Injector Current Source Required**
- **64-Pin DIPs with 50-mil Pin Centers**

Description

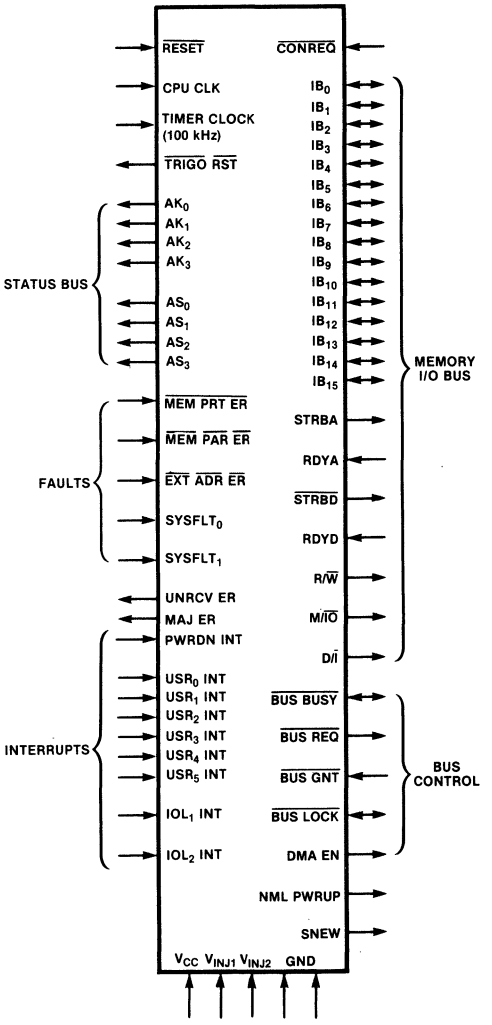
The F9450 microprocessor, in a single chip, completely implements MIL-STD 1750A (Notice 1) Instruction Set Architecture. This microprocessor is currently being developed as the heart of a high-performance processor family for commercial and military applications requiring high-speed, sophisticated, real-time processing.

Utilizing 16-bit architecture, the F9450 provides 16 user-accessible general-purpose registers and performs floating point operations on-chip. The I^3L-II bipolar VLSI technology affords static operation with 200 ns bus cycle times, low-power Schottky input/output, inherent radiation tolerance (1 x 10^5 rads), and operation at 20 MHz over the full military temperature range.

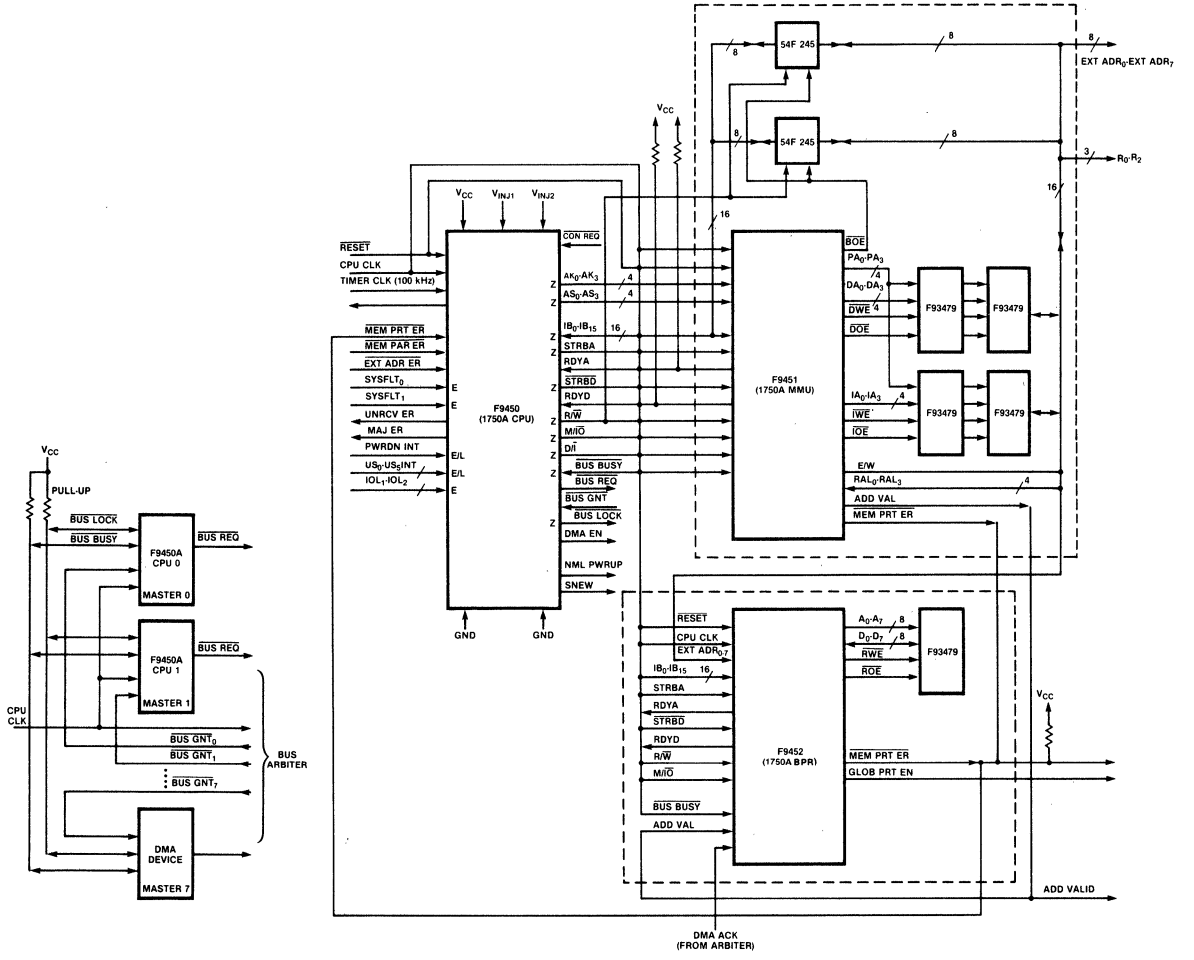
Real-time processing is achieved through advance design and architecture, incorporating two programmable timers, a complete 16-level interrupt processor, and a comprehensive fault handler on the chip.

Several support circuits and systems can provide additional capability. These include the F9443 Math Co-processor, options for MIL-STD 1750A built-in functions, user-programmable functions, and IEEE floating point; the F9446 Dynamic Memory Controller; the F9451 Memory Management Unit, providing memory-mapped expansion to 1M words; and the F9452 Block Protect RAM. A multi-user development system (FS-I) has been developed, as well as EMUTRAC™, which offers real-time system emulation and debugging.

Signal Functions



6



Instruction Execution Times (μs) - 50 ns CPU Clock Period

	Single Precision Integer	Double Precision Integer	Floating Point	Extended Floating Point
Register Add/Sub	0.2	0.8	4.5	5.75
Register Multipy	1.85	5.75	5.6	12.4
Register Divide	4.7	12.0	9.8	21.5
Load Direct	0.6	1.25	1.25	1.3
Branch	Taken = 0.75 μs		Not Taken = 0.2 μs	

External Arbitration

In an external arbitration of a multi-processor system, the bus arbiter receives requests ($\overline{\text{BUS REQ}}$) from each of the bus masters and issues the bus grant ($\overline{\text{BUS GNT}}$) to the highest priority bus master. That bus master will acquire the bus only if it is not locked ($\overline{\text{BUS LOCK}}$ not active).

Memory Protection and Management

In this configuration, the F9450 is connected to the MMU and the BPR. The MMU consists of the F9451, the maps for instruction and data (four F93479s), and two bidirectional drivers. The BPR consists of the F9452 and protection tables for CPU and DMA modes (F93479). Protection errors from the MMU and BPR units are wired-OR to the CPU $\overline{\text{MEM PRT ER}}$ input.

FAIRCHILD
A Schlumberger Company

F9451
Memory Management Unit

Advance Product Information | Microprocessor Product

Description

The Fairchild F9451 Memory Management Unit (MMU) provides the logical-to-physical address translation for instructions and operands in the MIL-STD 1750A configuration. The MMU serves to expand to one million words the direct addressing of the F9450 CPU, and provides protection in logical space units of 4K-word pages for access key, write, and execute instructions. Figure 1 illustrates the addressing structure, and figure 2 shows a block diagram of the MMU.

- **Logical-to-Physical Address Translation for Instructions and Operands**
- **One Million Word Addressing Space**
- **Protection in 4K-Word Pages for Access Key, Write, and Execute**
- **Instruction and Operand Maps**
- **I^3L® Technology**
- **Operating Temperature Range of from −55°C to +125°C**
- **Radiation-Tolerant Technology**
- **64-Pin DIP or Optional Leaded Chip Carrier**

6

Figure 1 Addressing Structure

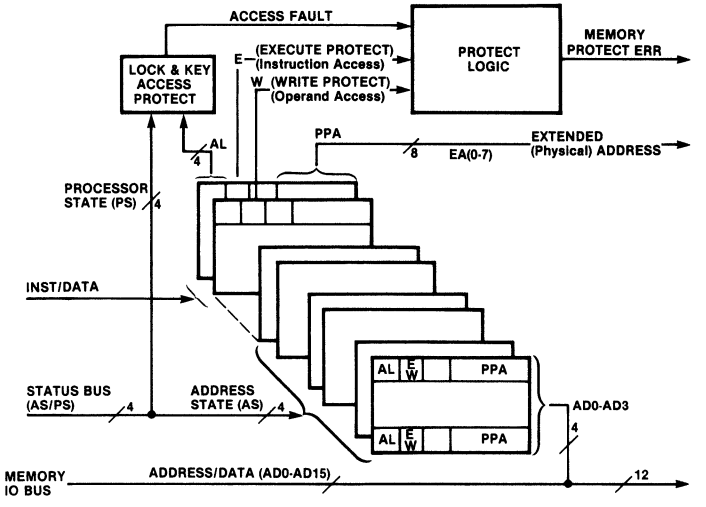
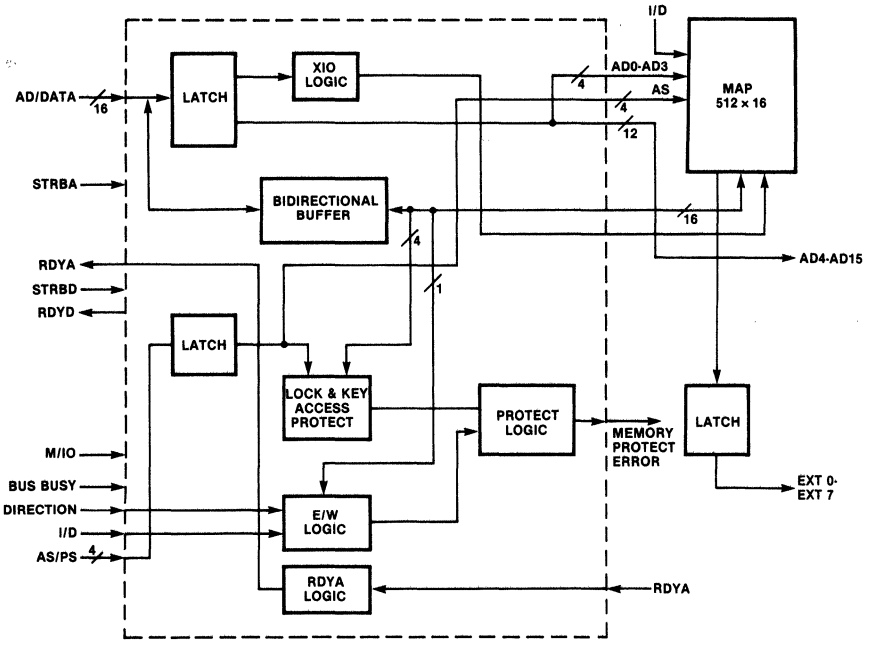


Figure 2 F9451 MMU Block Diagram



FAIRCHILD
A Schlumberger Company

F9452
Block Protect RAM

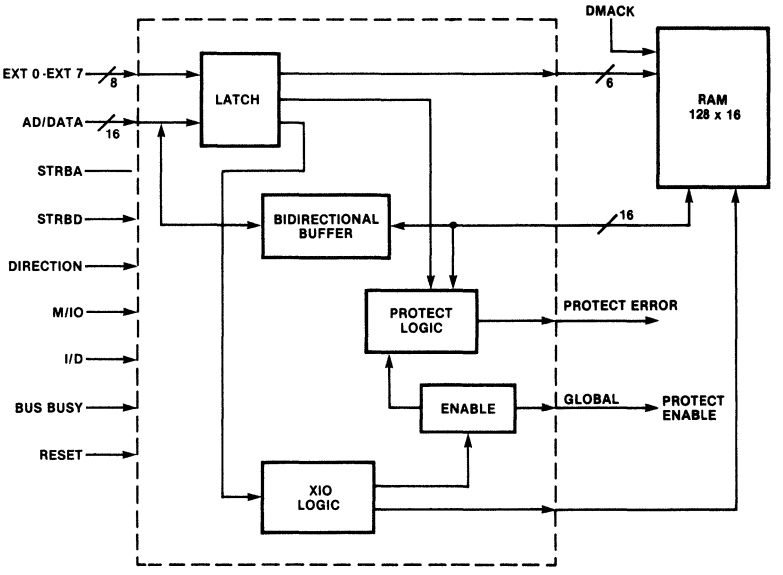
Advance Product Information Microprocessor Product

Description

The Fairchild F9452 Block Protect RAM (BPR) unit provides write protection in physical memory for the CPU and DMA in blocks of 1K words. The BPR also provides global write protection from initialization until enabled. Figure 1 is a block diagram of the BPR.

- **I^3L® Technology**
- **Operating Temperature Range From −55°C to +125°C**
- **Radiation-Tolerant Technology**
- **64-Pin DIP, or Optional Leaded Chip Carrier**

Figure 1 F9452 Block Diagram



® I^3L is a registered trademark of Fairchild Camera and Instrument Corp.

FAIRCHILD
A Schlumberger Company

F9470
Communication and Console Controller

Microprocessor Product

Description

The Fairchild F9470 Communication and Console Controller is an LSI MOS device that provides the Fairchild F9445 16-bit I^3L® microprocessor with virtual console control and programmed serial I/O via a pair of asynchronous communication ports.

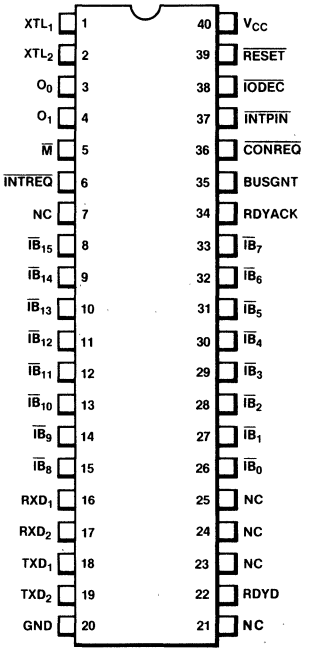
The F9470 provides a variety of useful console functions, including examine and deposit to memory and accumulators, jump to a specified location, and trace the F9445 instruction execution.

The F9470 operates in two modes: console control and I/O service. In the console mode, all communication with the F9445 is controlled by the F9470, which interprets user commands, requests the appropriate information from the F9445, and then outputs it to the operator's terminal.

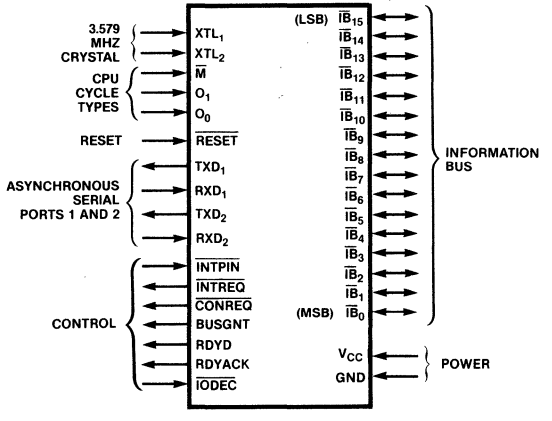
In the I/O service mode, the F9470 acts as a serial I/O controller, interfacing the serial I/O devices to the F9445 through device codes 10-13 and 77. The console commands are not available while in the I/O service mode; all I/O in this mode is programmed through the F9445.

- **Accesses Microprocessor Internal Registers**
- **Two Asynchronous Serial Ports**
- **Allows CRT Terminal to Operate as a Console for an F9445 System**
- **40-Pin DIP Requiring Single +5 V Power Supply**
- **NMOS Technology**

Connection Diagram



Signal Functions



Serial I/O

The F9470 has two asynchronous serial input/output ports. These ports can operate at 110, 300, 1200, 1800, or 2400 baud; port 1 can also operate at 4800 baud (refer to table 1 for baud rate selection codes and restrictions). The baud rate of both ports must be initialized through port 1. The baud rate of port 1 is set by typing a carriage return, <RETURN>, which should be the first key entered after power-up or after issuing the reset baud rate (S) command.

If the baud rate selected for port 1 is 4800, port 2 is disabled and the console mode is entered. If the baud rate of port 1 is 110, the port 2 baud rate is set to 110 and the console mode is entered. If the baud rate of port 1 is between 300 and 2400, the F9470 displays the prompt

```
PEP
+
```

indicating that the baud rate code for port 2 must be entered by the user. These codes are shown in table 1.

The baud rate of port 2 must be less than or equal to the baud rate of port 1, and certain combinations are not allowed (as indicated in the table). If the baud rate code entered for port 2 is incorrect, the F9470 again displays the PEP + prompt.

Console Mode Operation

Once the baud rates have been set, the F9470 enters the console mode (CM), as indicated by a * prompt. In the CM, the F9470 executes the eight console mode commands described in table 2. In response to the A, C, and E interrogative commands, the F9470 requests a console operation from the F9445. The F9445 executes that operation under the control of the F9470 and returns a result to the F9470, if appropriate. Leading octal operands are required with C and E commands. (Execution of interrogative commands halts the F9445.)

The J and R commands cause the F9445 to start program execution at a specified memory location. The J command transfers the F9470 from the CM to the I/O service mode (IOSM), while the R command leaves the F9470 in the CM. The J command causes the F9470 to transfer to the IOSM without affecting F9445 status.

The T command single-steps the F9445 under control of the F9470, and displays the CPU registers after each instruction. This command halts the F9445.

While executing the R and T commands, the F9470 does not respond to I/O instructions from the F9445; all I/O read operations (including tests of device flags) return a zero.

Table 1 Baud Rate Selections and Restrictions for Serial I/O

Baud Rate Codes for Port 2

Code	Baud Rate Selected
0	Disabled
1	110
2	300
3	1200
4	1800
5	2400

Baud Rate Options

Port 1 Baud Rates	Valid Baud Rates for Port 2					
	None	110	300	1200	1800	2400
110		x				
300	x		x			
1200	x		x	x		
1800	x		x		x	
2400	x	x	x	x		x
4800	x					

110 baud automatically sets port 2 to 110
4800 baud automatically disables port 2

Table 2 F9470 Console Mode Commands and Active Keys

Command	Function	Description
		COMMANDS
A	Display Accumulators	Displays in octal the following registers: PC, AC0, AC1, AC2, AC3, SP, and FP.
n,vC	Change Accumulator	Deposits octal value v into register n, where n is n: 0 1 2 3 4 5 Corresponding Register: AC0 AC1 AC2 AC3 SP FP
x:v	Deposit Memory	Deposits octal value v into memory location x.
xE	Examine Memory	Opens and displays the contents of memory location x, allowing octal numbers to be entered. Memory locations may be closed by: <RETURN> which opens the next memory location ^ (caret) which opens the previous memory location <ESC> which returns to the monitor
xJ	Jump	Loads F9445 PC with x, starts program execution, and transfers to the IOSM. When x = 0, transfers to the IOSM, and does not affect F9445 status.
xR	Run	Same as Jump, except that F9470 remains in the CM. When x = 0, transfers to the IOSM, and does not affect F9445 status.
S	Set Baud Rate	Resets F9470. The <RETURN> following S initialized the baud rate of port 1.
nT	Trace	Traces n octal steps of F9445 program execution, displaying CPU registers after each instruction, in the form: Instruction Address Instruction AC0 AC1 AC2 AC3 SP FP Before tracing a program, set the program counter to the appropriate location, using the xE command. Trace defaults to single-step, if n is omitted.
		ACTIVE KEYS
BACKSPACE		Deletes the previously-typed character.
BREAK		Transfers the F9470 from the IOSM to the CM. Does not affect F9445 program execution except that the F9445 I/O instructions are no longer interpreted by the F9470.

I/O Service Mode Operation

In the I/O service mode, the F9470 acts as a serial I/O controller, interfacing two serial I/O ports to the F9445 using device codes 10 through 13. Executing the J command changes the F9470 from the CM to the IOSM and causes the F9445 to start program execution.

In the IOSM, the F9470 communication format is 1 start bit, 8 data bits, and 2 stop bits. Any parity bit should be set to 1. Table 3 lists the F9445 instruction that the F9470 recognizes when in the IOSM. These commands are a subset of the F9445 instructions described in the F9445 data sheet. When in the I/O service mode, the F9470 optionally responds to the device code 77 control instructions (defined in table 4). These device codes (10, 11, 12, 13, and 77) are octal values of the six least significant instruction bits (refer to table 5).

Pressing the BREAK key at any time transfers the F9470 from the IOSM to the CM. The BREAK does not affect the F9445 program execution; therefore, execution continues until a console command is executed. The F9470 does not interpret F9445 I/O commands after the BREAK is pressed. Because the execution speeds of the F9445 and the F9470 differ, certain programming considerations must be made when using the F9470.

1. The busy flag of the F9470 input ports is not visible to the programmer, hence the SKPBN and SKPBZ instructions should not be used with device codes 10 or 12.

2. After clearing a device-done flag, the F9470 requires time to clear the associated interrupt request. This time, which depends on the baud rate, varies from 10-100 microseconds. When performing interrupt-driven I/O, add a delay between the clear interrupt instruction and the next F9445 interrupt enable instruction. This delay ensures that the previously serviced request is cleared before the F9445 interrupt flag is re-enabled.

   A simple method of adding the required delay is to insert a no I/O (NIO) instruction between the interrupt clear instruction (e.g., DIAC O,TTI) and the subsequent interrupt enable (INTEN) instruction.

3. The F9470 interrupt mask bits are set by applying power to the board or executing an IORST instruction. To perform interrupt-driven I/O with the F9470, the mask bits must first be cleared with an MSKO instruction.

Table 3 F9470 I/O Service Instructions

Instruction	Description
DIAx ACC,DEV	Data In from A
DOAx ACC,DEV	Data Out from A
NIOx DEV	No I/O; Used to Start or Clear a Device
SKPBN DEV	Skip if Busy = 1
SKPBZ DEV	Skip if Busy = 0
SKPDN DEV	Skip if Done = 1
SKPDZ DEV	Skip if Done = 0

NOTES
If x = S (start), set busy flag, clear done.
If x = C (clear), clear busy flag, set done.
ACC = Accumulator 0, 1, 2, or 3.
DEV = Device Codes = 10, 11, 12, 13.
Note that Busy is not defined for input devices (TTI and PTR); hence, SKPBN and SKPBZ should not be used with these devices.

Table 4 F9470 Interrupt Control Commands

Instruction	Description
IORST	Clears all busy and done flags and sets all interrupt disable flags (disabling interrupts).
MSKO ACC	Enables or disables device interrupts by clearing or setting the interrupt disable flag in the device. The interrupt disable flag of each device is associated with a specific data line, and is set if its mask bit is 1, cleared if 0.
INTA ACC	Reads device code of highest priority device that is requesting an interrupt. The 6-bit code is loaded into ACC bits 10-15. All 16 bits are set to 0 if no device is interrupting.

Interrupt Disable Bits for I/O Devices

IB Bits	8	9	10	11	12	13	14	15
Mnemonic				PTR		PTP	TTI	TTO
Function				CH2 In		CH2 Out	CH1 In	CH1 Out

Table 5 Instruction Device Codes for IOSM

Device Code	Mnemonic	Description	Action
10	TTI	Teletype In	Input on Channel 1
11	TTO	Teletype Out	Output on Channel 1
12	PTR	Paper Tape Reader	Input on Channel 2
13	PTP	Paper Tape Punch	Output on Channel 2
77	CPU	CPU	

Hardware Interface

The F9470 interfaces to the F9445 through the 16-bit information bus (IB), the processor status lines ($\overline{M}$, O_1, O_0), and the processor control lines (RDYA, RDYD, BUSGNT, $\overline{CONREQ}$, and $\overline{INTREQ}$). All data transfers between the F9445 and the F9470 occur on the IB. These transfers are controlled by the three bus wait lines (RDYA, RDYD, and BUSGNT). Refer to figure 1.

The F9470 requests interrupt and console service by asserting $\overline{INTREQ}$ and $\overline{CONREQ}$, respectively. The F9470 determines the processor cycle type (e.g., read console code, or read console data) during console operations by sensing the $\overline{M}$, O_1, and O_0 lines. (For more information on F9445 data transfers, see the F9445 data sheet.) The F9470 uses standard serial line driver/receivers (1488/1489) and requires a 3.57954 MHz parallel resonant crystal.

The F9470 outputs use open collectors with internal 6 kΩ pull-up resistors. The $\overline{CONREQ}$, $\overline{INTREQ}$, BUSGNT, and RDYD outputs may interface directly to the F9445 if the F9470 drive characteristics are not exceeded. All F9470 inputs, except XTL_1 and XTL_2, have internal 6 kΩ pull-up resistors.

The F9470 requires external logic that denies RDYA and asserts $\overline{IODEC}$ during an I/O instruction to devices 10-13 and optionally 77. There are three components to this decoding logic, as shown in figure 1:

1. An LS533 latches the upper four device code bits ($\overline{IB}_{10}$-$\overline{IB}_{13}$) using STRBA. This data is typically available elsewhere in an F9445-based system.

2. The LS138 decodes the I/O instruction device code and produces an output if the instruction addresses devices 10-13. Output also occurs with DS77 if J2 is installed.

3. The LS74 delays the address phase of all I/O instructions for two clocks to provide the delay needed by the LS138.

Jumper Options

The three jumpers, J1, J2, and J3, tailor the F9470 to a particular system. In response to an interrupt acknowledge (INTA) instruction, the F9470 samples interrupt priority in ($\overline{INTPIN}$). When low, $\overline{INTPIN}$ indicates that no higher priority I/O device has a pending interrupt request, and that the F9470 should place its interrupt code on the IB. The $\overline{INTPIN}$ signal should be driven by the $\overline{INTPO}$ of the next higher priority device, if any; otherwise, install J1.

The J2 circuit allows the F9470 to respond to device code 77 instructions (INTA, MSKO, and IORST). Remove J2 when the F9470 is used in a system with other I/O devices that demand fast response to device code 77 instructions.

The J3 circuit enables the F9470 decode logic. With J3 removed, the F9470 does not respond to the F9445 I/O instructions.

DC Characteristics

The dc characteristics of the console controller are provided in table 6 (T_A = 0°C to 70°C, V_{CC} = 5 V ± 10%, I/O Power Dissipation = 100 mW).

Figure 1 F9470 System Interface

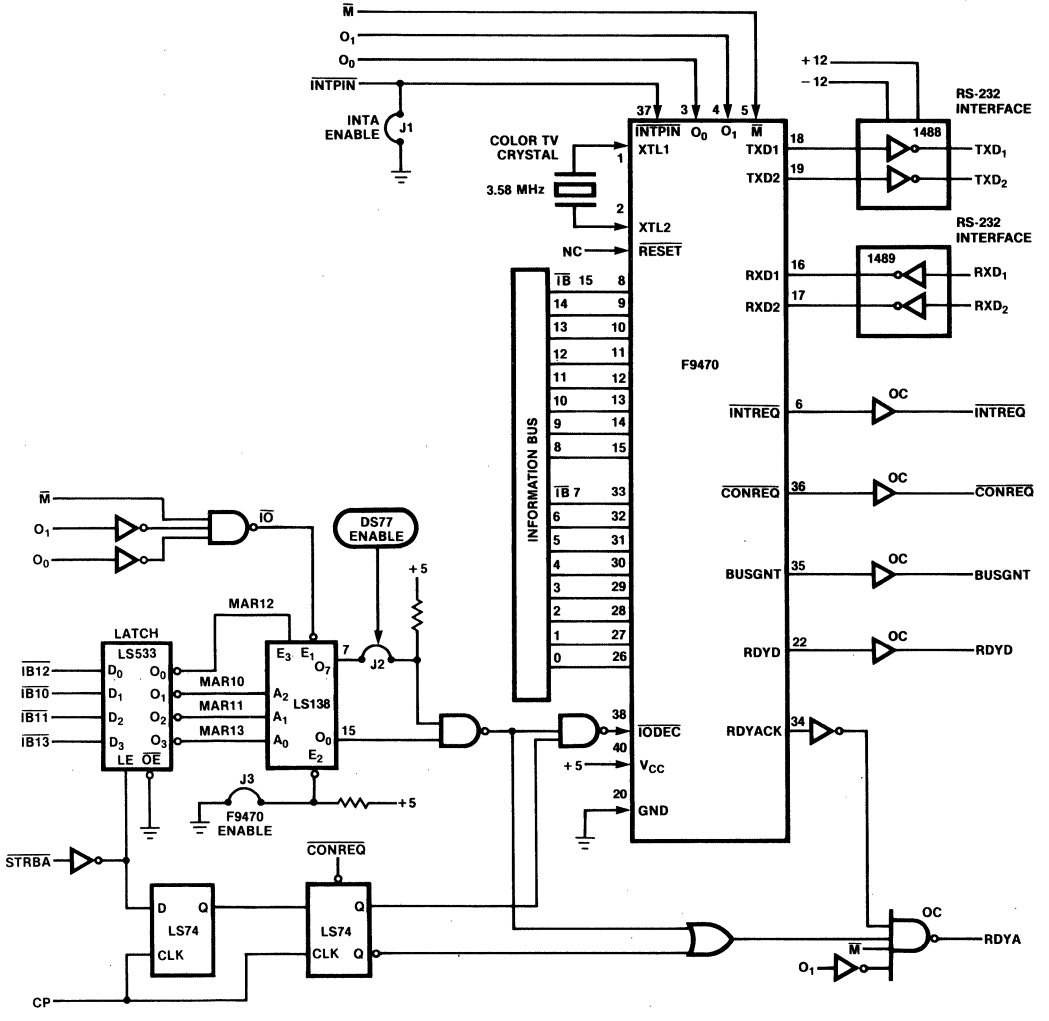


Table 6 Console Controller DC Characteristics

Symbol	Parameter	Min	Max	Unit	Test Conditions
I_{CC}	Power Supply Current	—	100	mA	Outputs Open
P_D	Power Dissipation	—	550.0	mW	Outputs Open
V_{IH}	Input High Voltage	2.0	5.8	V	
V_{IL}	Input Low Voltage	−0.3	0.8	V	
I_{LOD}	Leakage Current (open drain ports)	−10.0	+10.0	mA	Pull-down, device off V_{OH} = 13.2 V
I_{OL}	Output Low Current	1.8	—	mA	V_{OL} = 0.4 V

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings, or under any conditions above those indicated in this document, is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Temperature (ambient) under bias
- F9470DC 0 °C, +70 °C
- F9470DM −55 °C, +125 °C

Storage temperature −55 °C, +150 °C

Voltage on all open drain pins −1.0 V, +13.2 V

Voltage on all other pins, with respect to ground −1.0 V, +7.0 V

Power dissipation 1.5W

Ordering Information

Part Number	Package	Temperature Range*
F9470DC	Ceramic	C
F9470DL	Ceramic	L
F9470DM	Ceramic	M

* C = Commercial Temperature Range 0 °C to +70 °C
L = Limited Temperature Range −40 °C to +85 °C
M = Military Temperature Range −55 °C to +125 °C

6

FAIRCHILD
A Schlumberger Company

Section 7
F16000 Microprocessor Family

General

The giant strides in semiconductor technology and the startling effect of large-scale integration on the price/performance enhancement of computing and storage elements is a familiar theme in technical, business, and financial literature. Despite these explosive advances, however, computer technology is still in its infancy. The rapidly accelerating pace of technology improvements is expected to continue through the 1980s, when the technology will reach the physical limits governing present materials and manufacturing techniques. New components and systems approaches will then continue the evolution, and the physics-based approach dealing with patterning, processing, and device modeling will be expanded to include information-based aspects of design and architecture.

As computing costs continue to rapidly decline, the number of applications is exploding. Computers have now permeated most facets of our lives—calculators, automobiles, appliances, telephones, banks, weapons, and even toys—and will continue to play an ever-increasing role. The greatest impediment in this trend has been and continues to be the software; software development and maintenance remain manpower-intensive and represent a growing percentage of total system cost. The high software cost and increased software demand have created a major problem for both the information processing and automation industries.

Sound engineering and management disciplines have been moderately successful in controlling software cost and quality. However, it has become apparent that new, more powerful hardware architectural features are necessary to more consistently enhance cost and quality.

The Fairchild F16000 16-bit microprocessor family, the CPU of which has a 32-bit internal structure, is designed to significantly reduce software costs while improving software quality by incorporating powerful features into its architecture. These architectural features include:

Addressing

Addressing and memory management have always been problems in software engineering. A great deal of designing and programming time, as well as computer time, is spent on address manipulation. In an attempt to reduce this problem, the number of addressing modes has proliferated; this, in turn, has led to increased programming difficulties, partly because certain addressing modes could only be used by some instructions. In addition, addressing has been limited to a small space (about 64K bytes), requiring complex, error-prone schemes for segmenting large programs. The F16000, however, provides 16 megabytes of uniform, unsegmented address space. Any data structure—word, double word, byte, or bit—may be addressed anywhere in the space by any instruction using any one of the 11 addressing modes. This eliminates overhead, errors associated with memory-segment address calculation, and software compatibility problems for programs written for different address spaces.

Virtual Memory

Along with addressing main memory, management of total system memory hierarchy (fast disk, removable disk, magnetic tape, etc.) has been expensive, as well as a source of error. The F16000 provides facilities for system programs to treat combinations of various storage media as a single large memory space; large programs may be written without concern for size. The programmer need not worry about main or secondary memory management, paging, segmentation, swapping segments in and out of main memory, or locating the necessary data; the F16000 memory management unit manages the peripheral memory. If the required data for the instruction being executed in the F16000 CPU is not available in the main storage, the CPU can abort the instruction and return to the state existing prior to the execution of that instruction. The CPU then requests the necessary data and is free to perform other tasks until the data is available for re-execution of the aborted instruction. All this is totally transparent to the programmer. Thus, the cost of extensive memory management routines and the potential associated errors are eliminated.

Symmetry

Exceptions to the rules, special cases, and arbitrary restrictions on the use of programming facilities contribute heavily to both cost and software errors. This is especially apparent in modern powerful machines, where perhaps a hundred instructions, together with ten addressing modes, several data types, and classes of registers, make it difficult to keep track of which memory or register instruction can or cannot use which addressing mode. The F16000 architecture is totally symmetric in terms of instructions, addressing, registers, and data-type handling. All instructions function in the same way for any of the registers in any addressing mode.

High-Level Language Support

High-level languages have played a major role in reducing software costs. For example, programmer productivity is about 1.5 to 4.5 instructions per man-day using low-level assembly language for a complex program. Productivity rises to 9 to 16 instructions per man-day for the same program using a high-level language. In addition, high-level languages are less cryptic and can be understood and modified more easily. In many cases, high-level language structures provide less error-prone software, thereby achieving a higher system integrity level. The F16000 architectural features previously discussed, together with the support of arrays, queues, stacks, and records in addition to primitive data types (bits, bytes, etc.) are well suited to the efficient use of high-level language processing and compilation.

Modularity

Modularity is the design of small, self-contained independent programs, called modules, that may be used in many different combinations to perform specific tasks. It impacts all aspects of software engineering. The extent of system support of modularity affects the development and maintenance of program libraries consisting of many different general-purpose and special modules, which are used to build a complete software system. This is especially important for read-only memory (ROM) software. The F16000 provides for absolute modularity by imposing no restrictions on the individual module codes. Programs written independent of other programs may be loaded anywhere in the address space and mixed with other programs in any order. The F16000 module table is then set up by the linking loader to point to the code, data, and linkage information with the modules. At execution time, the F16000 registers point to appropriate code and data areas, thus eliminating the need for relocation, initialization, or other overhead functions.

Slave Processors

Specialized functions, such as floating point processing, memory management, fast Fourier transfer, etc., can be incorporated into auxiliary processing elements, such as chips, to replace reams of software that would otherwise reside in the computer memory. In the F16000, the slave processor instructions provide an extension of the main CPU instruction set to augment the expansion of system capabilities in hardware.

System Protection

Devising hardware techniques for detecting software errors and preventing propagation of these errors to sensitive system parts is relatively new in microprocessor design. In the F16000, for example, privileged instructions that can only be executed in the supervisory mode bar the lower-level applications programs from access to certain system resources.

Future Expansion

Software is a major part of an integrated approach to product and market development in a continuously changing environment involving product lines, services, and customer requirements. End-user software is therefore viewed as an experience base, a trade tool; the software expenditure is an investment to be conserved. This can best be accomplished by protecting the computer architecture from early obsolescence.

The open-ended architecture of the F16000 provides for the following expansion:

—Increasing direct addressing to four billion bytes of real memory space

—Extending the instruction set by adding slave processors

—Expanding to 32-bit computers, since the base architecture, including registers and internal data paths, is 32 bits wide

—Increasing virtual memory space to the billion or trillion byte range by adding a virtual memory translator and memory management chips

Architecturally, slave processors, including virtual memory translators, are considered part of the central processor. The F16000 slave processor instruction sets are intentionally designed as extensions of the CPU instructions so that, as the expected higher levels of semiconductor integration are achieved, F16000 devices can contain various degrees of extended capability on one chip.

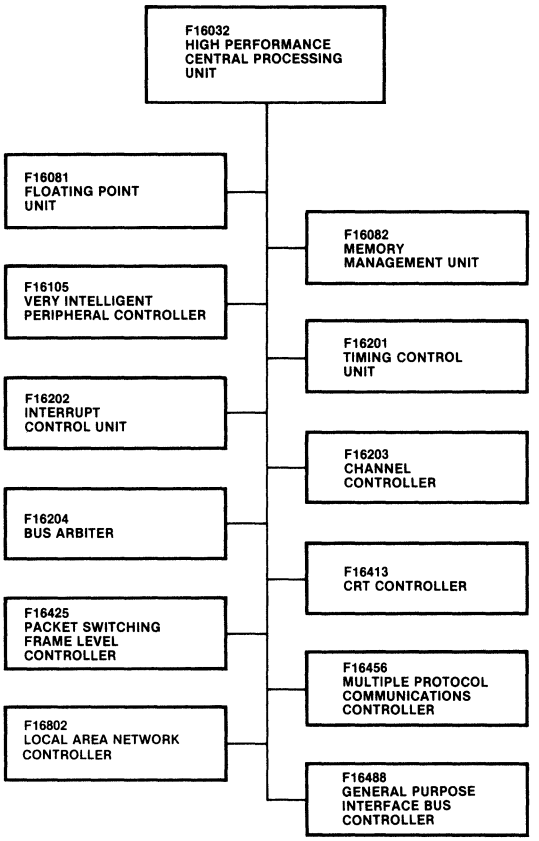
Organization

Figure 7-1 illustrates the organization of the F16000 family.

Descriptions

Following is data that describes the members of the F16000 microprocessor family.

Figure 7-1 F16000 Microprocessor Family Organization



FAIRCHILD
A Schlumberger Company

F16032
High-Performance CPU

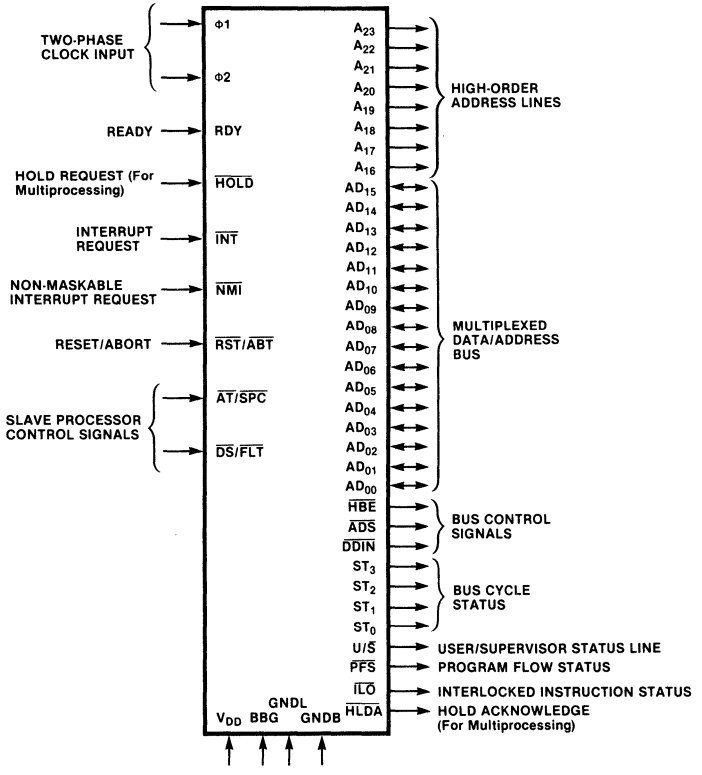
Advance Product Information | Microprocessor Product

Description

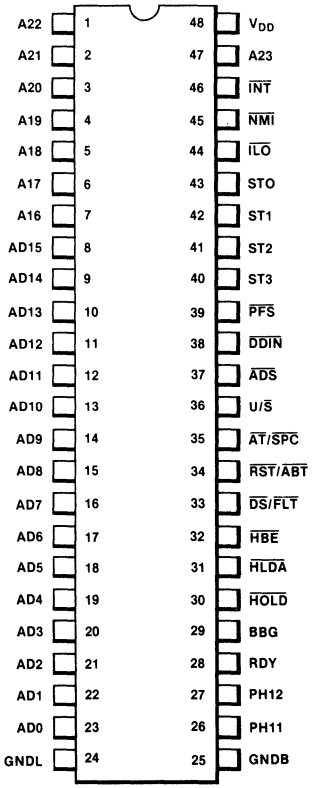
The F16032 central processing unit (CPU) acts as the highest performance CPU for the F16000 microprocessor family. It is designed to provide optimal support to microprocessor users who need a large addressing space for large programs or data structures. The F16000 architecture allows very efficient compilation for large programs generated and maintained in high-level languages, while remaining easy to program at the assembler level for optimizations. Figure 1 is a block diagram of the F16032 CPU.

- **32-Bit Architecture and Implementation**
- **16-Mb Uniform Addressing Space**
- **Two-Address Architecture**
- **High Degree of Symmetry in Instruction Set**
- **Addressing Modes Designed to Support High-Level Language Accesses to Variables**
- **High-Speed N-Channel MOS Technology**
- **Single +5 V Power Supply**
- **48-Pin Dual In-Line Package (DIP)**

Signal Functions

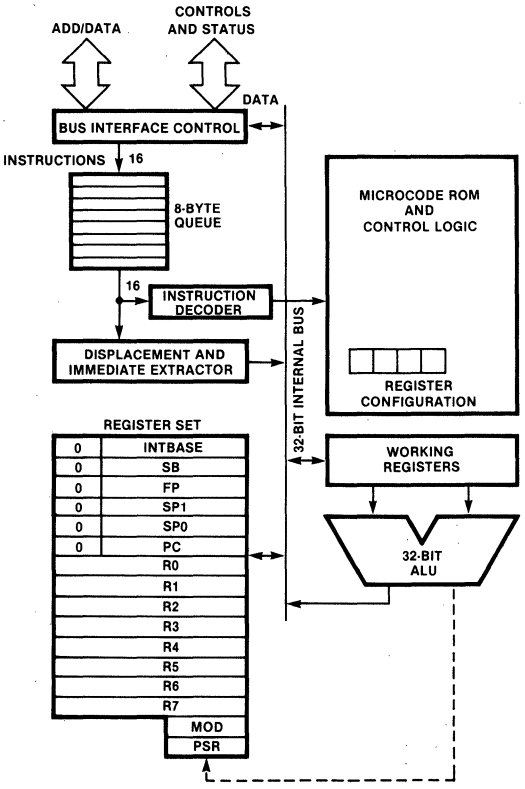


Connection Diagram



7

Figure 1 F16032 Block Diagram



Registers

The eight dedicated and eight high-speed general storage registers of the F16032 are illustrated in figure 2.

Instruction Set

Table 1 is a summary of the instruction set for the F16032, and figure 3 shows the general instruction format.

Figure 2 F16032 Dedicated and General Storage Registers

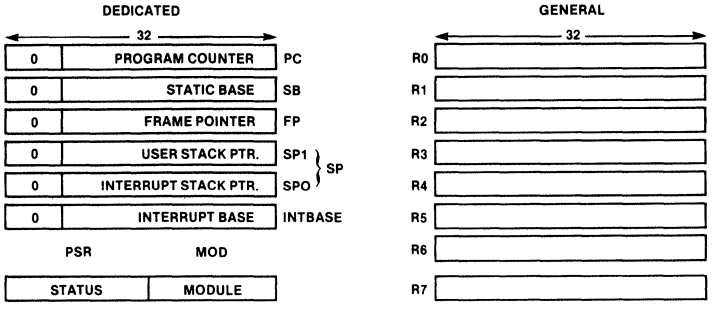


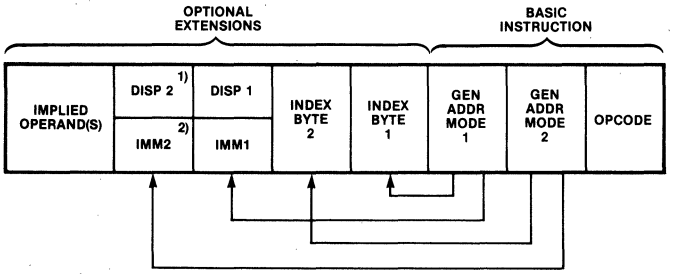
Table 1 F16032 Instruction Summary

Mnemonic	Meaning
ABS	Absolute Value
ACB	Add Compare and Branch
ADD	Add
ADDC	Add with Carry
ADDP	Add Packed Decimal
ADDQ	Add Quick Integer
ADDR	Calculate Address
ADJSP	Adjust Stack Pointer
AND	And
ASH	Arithmetic Shift
B	Conditional Branch
BIC	Bit Clear
BICPSR	Bit Clear in PSR
BISPSR	Bit Set in PSR
BPT	Breakpoint Trap
BSR	Branch to Subroutine
CASE	Case Branch
CBIT	Clear Bit
CHECK	Check Index
CMP	Compare
CMPM	Compare Multiple
CMPQ	Compare Quick Integer
CMPS	Compare String
COM	Complement
CVTP	Convert to Bit Pointer
CXP	Call External Procedure
CXPD	Call External Procedure with Descriptor
DFI	Divide Extended Integer
DIV	Divide
ENTER	Enter New Context
EXIT	Exit Context
EXT	Extract Field
EXTS	Extract Field Short
FFS	Find First Set Bit
FLAG	Flag Trap
IBIT	Invert Bit
INDEX	Calculate Index
INS	Insert Field
INSS	Insert Field Short
JSR	Jump to Subroutine
JUMP	Jump
LMR	Load MMU Register
LPR	Load Processor Register
LSH	Logical Shift
MEI	Multiply Extended Integer
MOD	Modulus of Periodic Function
MOV	Move
MOVM	Move Multiple
MOVQ	Move Quick Integer
MOVS	Move String
MOVSU	Mover Supervisor to User
MOVUS	Mover User to Supervisor
MOVX	Sign Extend
MOVZ	Zero Extend
MUL	Multiply

Table 1 F16032 Instruction Summary (Cont'd.)

Mnemonic	Meaning
NEG	Negate
NOP	No Operation
NOT	Not
OR	Or
QUO	Quotient
RDVAL	Read Address Validate
REM	Remainder
RESTORE	Restore General Registers
RET	Return from Subroutine
RETI	Return from Interrupt
RETT	Return from Trap
ROT	Rotate
RXP	Return from External Procedure
S	Set on Condition
SAVE	Save General Registers
SBIT	Set Bit
SETCFG	Set Configuration
SKPS	Skip String
SMR	Store MMU Register
SPR	Store Processor Register
SUB	Subtract
SUBC	Subtract with Borrow
SUBP	Subtract Packed Decimal
SVC	Supervisor Call
TBIT	Test Bit
WAIT	Wait
WRVAL	Write Address Validate
XOR	Exclusive OR

Figure 3 F16032 General Instruction Format



DC Characteristics

The dc electrical characteristics of the F16032 CPU are presented in table 2.

Table 2 DC Electrical Characteristics

$T_A = 0$ to 70 °C, $V_{DD} = 5V \pm 5\%$, GND = 0V

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
V_{IH}	Logical 1 Input Voltage	2.0		$V_{DD} - 0.5$	V	
V_{IL}	Logical 0 Input Voltage	−0.5		0.8	V	
V_{CH}	Logical 1 Clock Voltage	$V_{DD} - 0.3$		$V_{DD} + 1.0$	V	Φ1, Φ2 Pins Only
V_{CL}	Logical 0 Output Voltage	−0.5		0.3	V	Φ1,Φ2 Pins Only
V_{OH}	Logical 1 Output Voltage	2.4			V	$I_{OUT} = -400\ \mu A$
V_{OL}	Logical 0 Output Voltage			0.45	V	$I_{OL} = 2$ mA
I_{ILS}	AT/SPC Input Current (low)			1.0	mA	$V_{IN} = 0.4V$, AT/SPC in Input Mode
I_{IL}	Input Leakage Current	−1.0		1.0	μA	$V_{IN} \leq V_{DD}$, All Inputs Except Φ1,Φ2, AT/SPC
I_{OL}	Output Leakage Current	−1.0		1.0	μA	$0 \leq V_{IN} \leq V_{DD}$
I_{DD}	Active Supply Current			300	mA	$I_{OUT} = 0$, $T_A = 0$ °C

Absolute Maximum Ratings

The absolute maximum ratings for the CPU are presented in table 3. These are stress ratings only, and functional operation at these ratings or under any conditions above those indicated in this data sheet is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Table 3 Absolute Maximum Ratings

Temperature Under Bias	0 °C, +70 °C
Storage Temperature	−65 °C, +150 °C
All Input or Output Voltages with Respect to GND	−0.5 V, +7.0 V
Power Dissipation	1.5 W

FAIRCHILD
A Schlumberger Company

F16081
Floating Point Unit

Advance Product Information

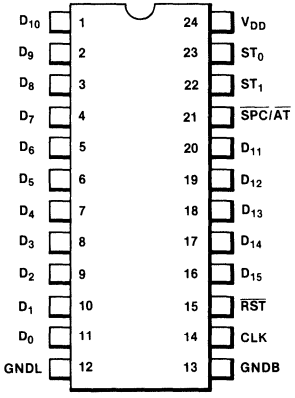
Microprocessor Product

Description

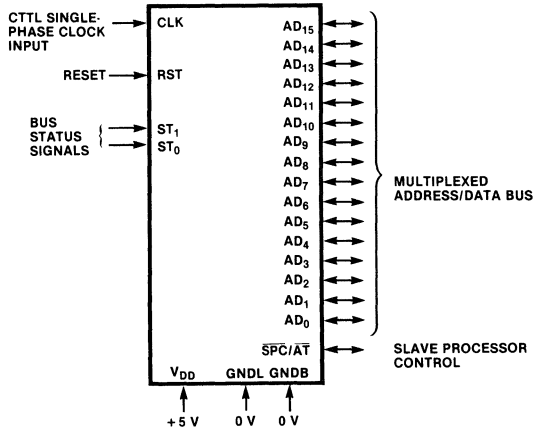
The F16081 Floating Point Unit (FPU) is a slave processor intended to augment the instruction set of the F16000 microprocessor family. The FPU implements a version the IEEE standard P754 floating point specification.

- **High-Speed Operation**
- **Single (32-Bit) and Double (64-Bit) Precision**
- **Selectable Rounding Modes**
- **Error Detection and Interrupt Generation**
- **NMOS Technology**
- **24-Pin Dual-In-Line Package (DIP)**
- **+5 V Power Supply**

Connection Diagram



Signal Functions



7

Instruction Summary

The instructions for the F16081 are summarized in table 1.

Table 1 F16081 Instruction Summary

Mnemonic	Description
ABSf	Absolute Value Floating Point
ADDf	Add Floating Point
CMPf	Compare Floating Point
DIVf	Divide Floating Point
FLOORfi	Floor Function
LFSR	Load Floating Point Status Register
MOVf	Move Floating Point
MOVFL	Move and Convert
MOVif	Move and Convert
MOVLF	Move and Convert
NEGf	Negate Floating Point
ROUNDfi	Round Function
SFSR	Store Floating Point Status Register
SUBf	Subtract Floating Point
TRUNCfi	Truncate Function

FAIRCHILD
A Schlumberger Company

F16082
Memory Management Unit

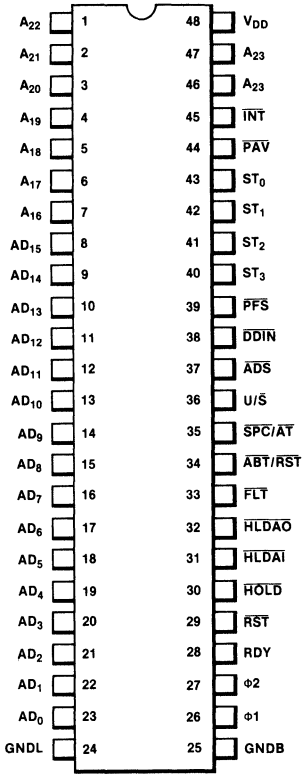
Advance Product Information | Microprocessor Product

Description

The F16082 Memory Management Unit (MMU) provides support for virtual memory management and program debugging when used with the F16000 microprocessor family. It is designed to relieve the microprocessor unit of burdensome tasks associated with memory management and provide address translation during program execution. The MMU converts virtual addresses issued by the MPU to physical addresses. Support is included to assist the operating system in implementing memory management policies. Memory protection is implemented by slave instructions that check the validity of a memory reference. The F16082 also permits easy implementation of a virtual machine in a debugging and in-system emulation environment. The MMU slave processor extends the memory management capabilities of the F16000 microprocessor family. Slave processor concepts allow potential software compatibility with future systems because the slave hardware is transparent to the software.

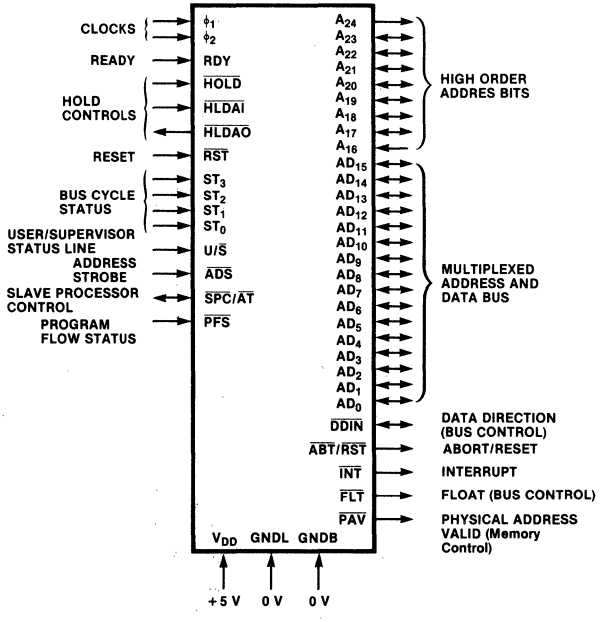
- **Dynamic Address Translation Using Memory Page Tables**
- **On-Chip Cache for the 32 Most Recently Used Memory Page Table Entries**
- **Virtual Memory**
- **Memory Protection**
- **Program Breakpointing**
- **Program Flow Tracing**
- **Virtual Machine Support**
- **High-Speed NMOS Fabrication**
- **48-Pin Dual-In-Line Package (DIP)**
- **Single +5 V Power Supply**

Connection Diagram



7

Signal Functions



Instruction Summary

The instruction commands for the F16082 are summarized in table 1.

Table 1 F16082 Instruction Summary

Mnemonic	Description
LMR	Load MMU Register
RDVAL	Read Validate
SMR	Store MMU Register
WRVAL	Write Validate

Absolute Maximum Ratings

These are stress ratings only, and functional operation at these ratings or under any conditions above those indicated in this data sheet is not implied. Exposure to the absolute maximum rating conditions for extended periods of time may affect device reliability, and exposure to stresses greater than those listed may cause permanent damage to the device.

Temperature Under Bias	0°C, +70°C
Storage Temperature	−65°C, +150°
All Input or Output with Respect to GND	−0.5 V, +7.0 V
Power Dissipation	1.5 W

FAIRCHILD
A Schlumberger Company

F16105
Very Intelligent Peripheral Controller

Advance Product Information

Microprocessor Product

Description

The F16105 Very Intelligent Peripheral Controller (VIPC) is a general-purpose device to be used either with the F16000 microprocessor family or as a stand-alone system element. The controller is easy to program in both assembly language and high-level language.

- **Upward Software Compatibility with the F16032 Microprocessor**
- **Internal 10-MHz Clock Speed with Instruction Prefetch**
- **4096-Byte ROM**
- **192-Byte Two-Port RAM**
- **64-Byte Scratchpad RAM**
- **32 Pins Individually Programmable as I/O**
- **Asynchronous Communication Interface Port**
- **Cascadable 16-Bit Timer and Event Counters**
- **Eight Vectored Interrupts**
- **Input/Output Processor (IOP) or Single-Chip Configurations**
- **Remote or Local Bus Configurations**
- **External Memory Access of 56K Bytes**

FAIRCHILD
A Schlumberger Company

F16201
Timing Control Unit

Advance Product Information

Microprocessor Product

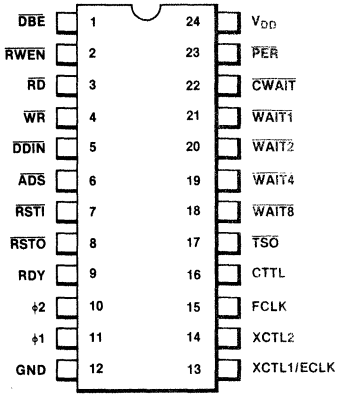
Description

The Fairchild F16201 Timing Control Unit (TCU) is a 24-pin Schottky component used with the Fairchild F16000 microprocessor family. It has four basic functions:

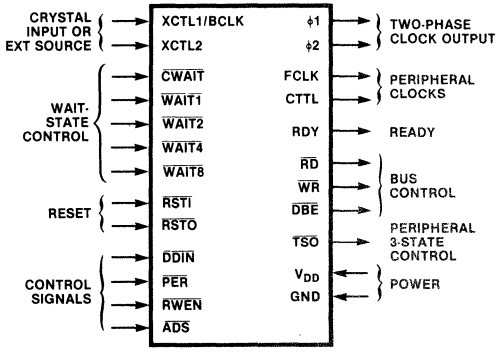
1. Provides two nonoverlapping clock phases for unbuffered use within the F16000 microprocessor family and provides a synchronous TTL output.
2. Provides the basic system read, write, and data-buffer-enable control signals.
3. Generates slow cycles compatible with the requirements of the older peripherals (those for which adding wait states is not sufficient).
4. Synchronizes the ready and reset inputs for the MPU.

- **Two Full-V_{DD} Swing Clock Outputs**
- **TTL Drive Capability on All Outputs Except Clock**
- **On-Chip Generator for F16000 Systems**
- **Bus Control Signals for F16000 Systems**
- **Support for Slow 8080 Peripherals**
- **Four Wait Inputs ($\overline{WAIT}$) to Force up to 15 Wait States**
- **Continuous Wait Input ($\overline{CWAIT}$) to Generate an Unlimited Number of Wait States**
- **Additional $\overline{CWAIT}$ Timing to Allow a Memory Cycle Hold and Subsequent Regeneration for System Arbitration or Memory Refresh**
- **Schmitt Trigger Reset Input, Internally Synchronized to Generate a Reset Output for the F16000 System**
- **Fast-Clock TTL Output with Twice the System Clock Frequency**
- **Frequency Range of From 0.2 MHz to 10.0 MHz**
- **Single +5 V Power Supply**

Connection Diagram



Signal Functions



7

Functional Description

The F16201 has five major elements (see figure 1): the oscillator and divide-by-2 circuit, the two-phase generator, the reset synchronization circuit, the wait-state generator, and the timing state counter and control signal generator (TSCCSG).

The oscillator and divide-by-2 circuit is connected to either an external crystal operating at twice the desired clock frequency or an external source by way of the XCTL1/ECLK pin. This circuit also generates the fast TTL clock (FCLK) with the crystal or ECLK frequency. A one-half-frequency output is created for the two-phase generator.

The two-phase generator provides two full-V_{DD} swing, nonoverlapping clock signals. Additionally, it generates a TTL clock (CTTL) and an internal clock to synchronize the other circuits of the chip.

The reset synchronization circuit synchronizes the reset-in input ($\overline{RSTI}$) to generate reset-out ($\overline{RSTO}$) with proper timing. The $\overline{RSTI}$ has a Schmitt trigger input.

Wait timing allows two different modes of operation, providing flexibility in the generation of the wait inputs. When the $\overline{CWAIT}$ or $\overline{WAIT}$ inputs are active, wait states are inserted. However, if $\overline{CWAIT}$ is used to create the wait, the $\overline{WAIT}$ inputs can be applied and are implemented following $\overline{CWAIT}$'s release. During a fixed wait count, $\overline{CWAIT}$ may also be taken to low, overriding the count and forcing a continuous wait to be entered.

The wait-state generator counts the number of wait states to be generated. A start pulse (generated by the TSCCSG circuit) is used to initiate the counting. While the counter is operating or the $\overline{CWAIT}$ input is low, the wait-state generator holds its wait output to the TSCCSG circuit active. The wait-state generator turns its ready output (RDY) to low when a start pulse is received from the TSCCSG. The RDY output returns to high when the wait signal is released.

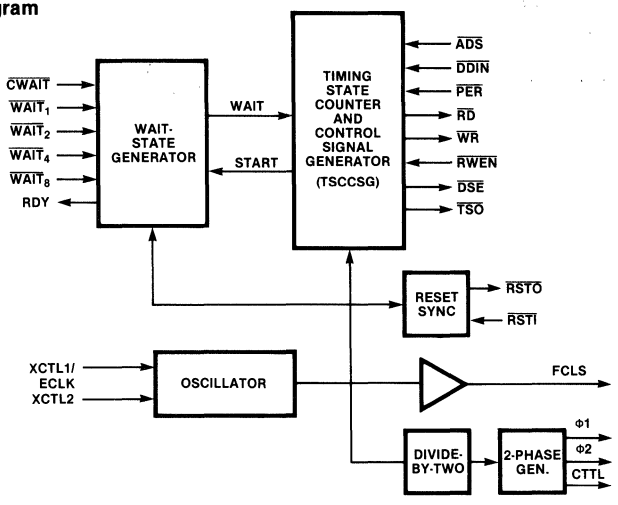
The timing state counter and control signal generator circuit keeps track of the timing state (T-state) within the MPU, generating the control signals accordingly. The arrival of the address strobe ($\overline{ADS}$) identifies the first T-state (T1) of a timing cycle. Input signals $\overline{DDIN}$ and $\overline{PER}$ are latched, and fast or slow, read or write cycles are generated. The TSCCSG circuit also extends a cycle, as directed by the wait line. The T-state output ($\overline{TSO}$) signal identifies the beginning of the second and last T-states of a timing cycle; it can be used to gate or clock external logic for synchronization.

Recommended Operating Conditions

The recommended operating ranges of the TCU are shown below.

Supply Voltage V_{CC}	4.75 Min., 5.25 Max V
Temperature T_A	0 °C, 70 °C

Figure 1 F16201 Block Diagram



FAIRCHILD
A Schlumberger Company

F16202
Interrupt Control Unit

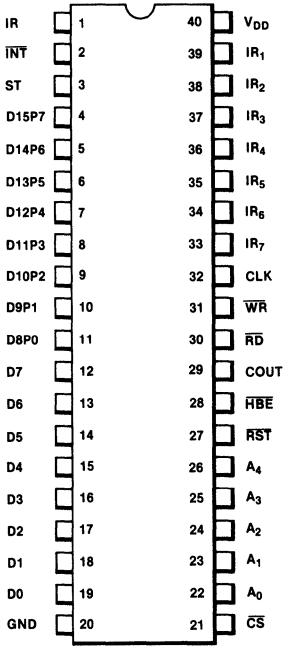
Advance Product Information | Microprocessor Product

Description

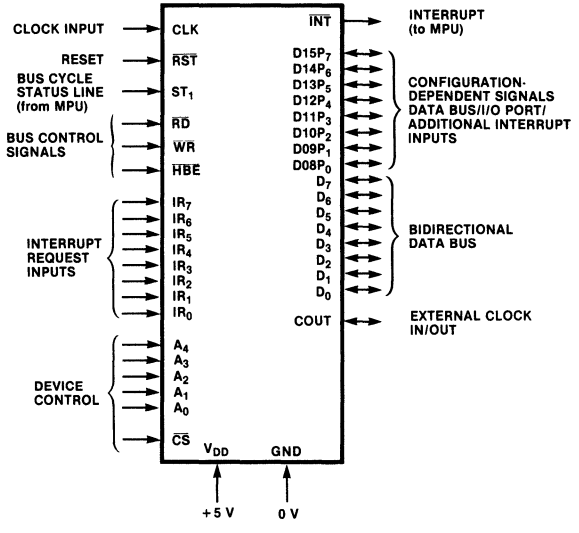
The F16202 Interrupt Control Unit (ICU) provides the F16000 microprocessor family with hardware support for prioritized, vectored interrupts and for a real-time clock.

- **16 Interrupt Sources, Cascadable to 256**
- **8 Hardware Interrupt Sources in 16-Bit Data Bus Mode**
- **Up to 16 Hardware Interrupt Sources in 8-Bit Data Bus Mode**
- **Optional 8-Bit Input/Output (I/O) Port when the 8-Bit Bus Mode is Used**
- **Five Optional Clock Outputs in the 8-Bit Bus Mode**
- **Two 16-Bit, dc-to-10 MHz Counters that Can Be Combined into a Single 32-Bit Counter**
- **Thirty-two 8-Bit Internal Registers Accessible as Pairs in the 16-Bit Bus Mode**
- **Software Interrupts**
- **Automatic Handling of Return from Interrupts**
- **Programmable Polarities and Level/Edge Selection for Each of the Hardware Interrupts**
- **Automatic Rotating Priority Mode**
- **NMOS Technology**
- **40-Pin Dual In-Line Package (DIP)**
- **Single +5 V Power Supply**

Connection Diagram



Signal Functions



7

FAIRCHILD
A Schlumberger Company

F16203
Channel Controller

Advance Product Information

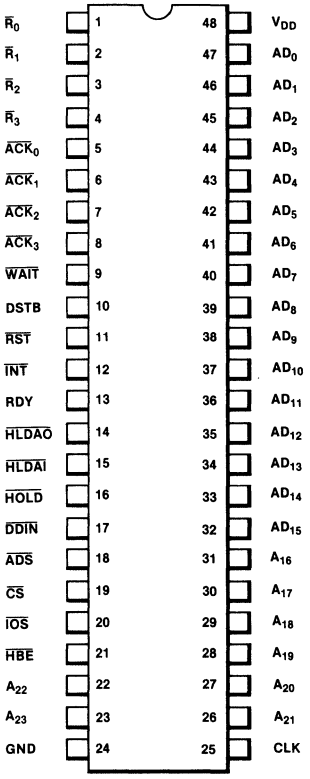
Microprocessor Product

Description

The F16203 Channel Controller is intended for use with the F16000 microprocessor family. It is a four-channel controller that can operate on a processor local multiplexed bus (via local mode selection) to support low-cost configurations. It can also operate with separate user-defined input/output (I/O) buses (via remote mode selection) when high performance is required.

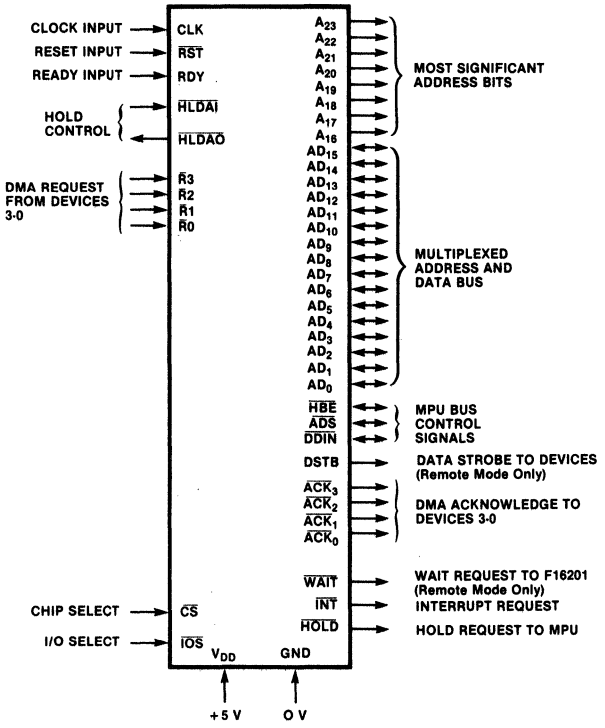
- **Up to Four Independent Channels**
- **Interfaces with the F16032 Central Processing Unit**
- **Integrated Operation with the F16202 Interrupt Control Unit**
- **Local/Remote (Single-/Multibus) Configurations**
- **Versatile Channel Commands**
- **Command Chaining**
- **Support for Memory-to-Memory and Device-to-Device Transfers**
- **8- and 16-Bit Devices**
- **NMOS Technology**
- **48-Pin Dual In-Line Package (DIP)**
- **Single +5 V Power Supply**
- **Maximum Data Rate of 5M Bytes Per Second**
- **Selectable Cycle Steal/Burst/Semiburst Transfers**

Connection Diagram



7

Signal Functions



Channel Command Summary

The basic channel commands for the F16203 are

DISABLE
VERIFY
SEARCH
TRANSFER AND SEARCH

Various command modifiers are appendable to the basic command and are summarized in table 1.

Table 1 F16203 Command Modifiers

Mnemonic	Description
AS	Word Assembly
AT	Auto Transfer
BT	Burst Type
D	Direct/Indirect Mode
DL	Destination Location
DT	Destination Type
DW	Destination Width
LP	Lock Priority
MN	Match/No Match
MNI	Match/No Match Interrupt Mask
PT	Priority Type
RQI	Request-While-Disabled Interrupt Mask
SE	Stop Enable
SL	Source Location
ST	Source Type
STI	Stop Function Interrupt Mask
SW	Source Width
TC	Transfer Complete
TCI	Transfer Complete Interrupt Mask
UW	Search Type

FAIRCHILD
A Schlumberger Company

F16204
Bus Arbiter

Advance Product Information | Microprocessor Product

Description

The F16204 Bus Arbiter manages heterogeneous multiprocessor systems that share a common bus.

- **Multiprocessor Environments**
- **Up to 32 Masters**
- **Selection of Arbitration Algorithm**
- **Encoded Arbitration Scheme**

FAIRCHILD
A Schlumberger Company

F16413
CRT Controller

Advance Product Information

Microprocessor Product

Description

The Fairchild F16413 CRT Controller (CRTC) operates within the F16000 microprocessor family for computer terminal, word processor, and monitor applications. The 64-pin, 5 V controller uses N-channel silicon gate technology. A number of features are programmable for easy adaptation to different display, synchronization, and screen formats.

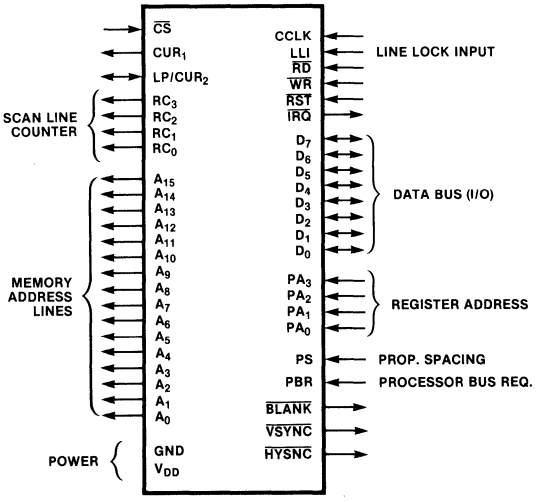
- **Programmable Display and Synchronization Formats**
- **Memory Addressing: Row/Column, DMA with Row Buffer, or Contiguous Linear Addressing**
- **Three Video Modes**
- **Three CRT Monitor Interfaces**
- **Programmable Window Location**
- **Programmable Status Field Location That Can Be Used to Provide a Vertical Split Screen**
- **Character Clock Rate to 10 MHz**
- **Maskable Interrupts: Line Zero, Vertical Blank, Smooth Scroll Complete, Programmable Row Interrupt, End of Scan Line**
- **Two Cursor Flags**
- **Double Width, Double Height Attributes**
- **Smooth Scrolling Forward or Reverse with Scroll Within Window or Its Inverse (Everything but the Window)**
- **Proportional Spacing**
- **Single +5 V Power Supply**

The F16413 CRTC features programmable display format for up to 256 characters per row, 128 rows per frame, and 16 rasters per row. It has a programmable format for horizontal and vertical sync pulse delay (front porch), sync pulse width, and scan delay (back porch).

Signal Descriptions

Table 1 describes the CRTC signals.

Signal Functions



7

Table 1 CRTC Signal Descriptions

Mnemonic	Name	Description
V_{DD}	Power	+5 V power supply.
GND	Ground	Commond ground.
A_0/RA_0-A_5/RA_5	Address Bus and Register Address	Lower six bidirectional address bus and register address pins.
A_6-A_{15}	Address Bus	Higher IO address bus output.
D_0-D_7	Data Bus	Bidirectional data bus.
$\overline{CS}$	Chip Select	Active low input to enable read/write of the internal registers during peripheral access.
$\overline{RD}$	Read Strobe	Input used to read data from the internal peripheral registers.
$\overline{WR}$	Write Strobe	Input used to write data to the internal peripheral registers.
$\overline{RST}$	Reset	An active low input to initialize the internal control and status registers.
$\overline{IRQ}$	Interrupt Request	An active low output that indicates one of the programmable interrupt conditions has occurred.
CCLK	Character Clock	Clock input to provide timing for synchronization and screen formatting, 10 MHz maximum.
RB_1	Read Row Buffer	Output, read row buffer number 1.
WB_1	Write Row Buffer	Output, write row buffer number 1.
RB_2	Read Row Buffer	Output, read row buffer number 2.
WB_2	Write Row Buffer	Output, write row buffer number 2.
$\overline{HOLD}$	Hold Request	Hold request output to the MPU.
$\overline{HOLDAI}$	Hold Acknowledge	Hold acknowledge input from the MPU when the CRTC works on the system memory; processor bus request when the CRTC works on dedicated video memory.
$\overline{HOLDAO}$	Hold Acknowledge	Hold Acknowledge output to a lower priority peripheral.
CUR_1	Cursor 1	Cursor 1 output.
CUR_2	Cursor 2	Cursor 2 output.

Mnemonic	Name	Description
HWS	Horizontal Window Start	Output indicating the first horizontal position of a window.
HWE	Horizontal Window End	Output indicating the last horizontal position of a window.
VWS	Vertical Window Start	Output indicating the first row of a window.
VWE	Vertical Window End	Output indicating the last row of a window.
PS	Proportional Space	Input causing an update of the RAM address counter; can also be used as a double wide attribute input.
DH	Double Height	Input for double height character attribute.
$\overline{\text{BLANK}}$	Blank	Active low output signal to turn off the monitor video during horizontal and vertical retrace.
$\overline{\text{HSYNC}}$	Horizontal Sync	Active low output signal to provide horizontal sync timing.
$\overline{\text{VSYNC}}$	Vertical Sync	Active low output signal to provide sync timing.
RC_0-RC_3	Raster Count Address	Outputs giving the current raster count value.

System Description

The F16413 CRTC can basically work in two system configurations, i.e., on a local bus with dedicated video memory or on the system bus with access to the main memory. In the latter case, external row buffers must be provided, which are being loaded during DMA operation with the characters of the next following row to be displayed.

There are 33 registers implemented on the F16413 for mode control, screen formatting, and display control. Each of these registers is individually addressable using the lower six address pins when chip select is active. Information can be read out from or written into the register via the 8-bit data bus.

Screen Format

The horizontally displayable dimension of the screen, horizontal front porch, sync pulse width, and back porch are programmable in character clocks. The vertically displayable dimension of the screen is programmable in number of rows. The vertical front porch, sync pulse width, and back porch are programmable in scan lines. The number of scan lines per row is 1 to 16.

Mode Control

The mode register determines the pin assignment for either system or remote bus operation. It is also used to select contiguous or row/column address, non-interlaced or interlaced video, attribute delay and external synchronization. It contains a reset control bit so the screen format may be reprogrammed anytime after a software reset.

Windowing and Scrolling

The status field may be used to provide a vertically split screen with the video field. The window feature of the F16413 allows a defined window anywhere in the video field, or splits the screen horizontally into two independent data fields. Using the split screen feature requires programming the number of horizontally displayed characters in each data field into the assigned registers.

The soft scroll control register is used to enable soft scroll, to select the area to be scrolled (either video field, window, or status field), and to select scroll rate and direction.

CPU/CRTC Memory Contention

The CRTC 3-states the address bus during the horizontal and vertical blanking intervals. An interrupt is provided at the beginning of each blanking interval.

Remote Bus Configuration

During the blanking interval the CPU is free to access the dedicated video memory without disturbing the display during the active video.

System Bus Configuration

The F16413 is accessing the system memory during DMA cycles to load external row buffers. If the user provides one external row buffer, the DMA operation starts with the first character of the first scan line of each row and the data is displayed immediately. In this case the RB and WB signals are both active during the first scan line of a row.

If two external row buffers are used, then one of them is being loaded with the data of the following row while the other one is sending its data of the present row to the CRT. Loading of the row buffers starts at the beginning of the first scan line of the previous row.

FAIRCHILD
A Schlumberger Company

F16425
Packet Switching Frame Level Controller (FLC)

Advance Product Information

Microprocessor Product

Description

The F16425 Packet Switching Frame Level Controller (FLC) is a member of the F16000 microprocessor family that controls the transmission and reception of data (message frames) in a network conforming to the international CCITT HDLC protocol for applications in terminals, network access controllers, and related equipment at level 2 (frame level). It implements X.25 LAPB and portions of X.75, SDLC, and HDLC. It can be used with most MOS microprocessor families.

The F16425 controller can be used specifically in data terminal equipment (DTE), data circuit terminating equipment (DCE), and network nodes (point-to-point, switched, or nonswitched systems). It uses all basic commands and responses—normal response mode (NRM) and asynchronous balanced mode (ABM) with five options. The CPU gives simple one-byte commands to initiate link setup, disconnect and information transfer.

- **8-Bit and 16-Bit CPU Compatibility**
- **High-Density NMOS SI-Gate Chip**
- **64-Pin DIP with 24 Registers at 8 Bits**
- **DC to 2.5 Mb/s**
- **One-Mbyte Direct Address Capability**
- **On-Board DMA to Transfer Messages to and from Memory**
- **Modem Interface Control Signals**
- **Programmable Address Field and Global Address**
- **Automatic Sequencing, Acknowledging, and Retransmission of Messages**
- **Automatic Frame Check Sequence Generation and Test**
- **TTL-Compatible**
- **Single +5 V Power Supply**
- **Separate Address and Data Bus**
- **8- or 16-Bit Bidirectional Data Bus**
- **Automatic Zero Insertion and Deletion for Transparency**
- **NRZ or NRZI Serial Data**
- **Programmable System Parameters**
  - **—Primary Timer (TI)**
  - **—Retransmission Counter (N2)**
  - **—Window Size from 1 to 127 Frames**
  - **—Buffer length from 16 to 2K Bytes**
- **Programmable Basic or Extended Control Field**
- **I-Field Residual Last Character**
- **X.25 LAPB, X.75 (Excluding Multilink), SDLC, HDLC, ADCCP**

Status can be monitored by a series of maskable interrupt conditions or by reading eight directly addressable status registers.

Figure 1 is a block diagram of the F16425 controller. Figure 2 shows how the F16425 interfaces with DTE and DCE.

Signal Descriptions

Table 1 describes the signals for the F16425 controller.

Signal Functions

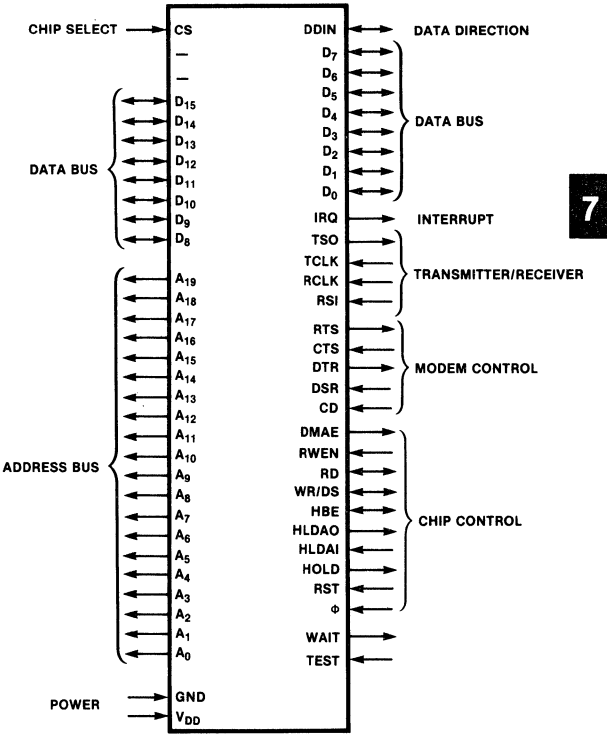


Figure 1 16425 Block Diagram

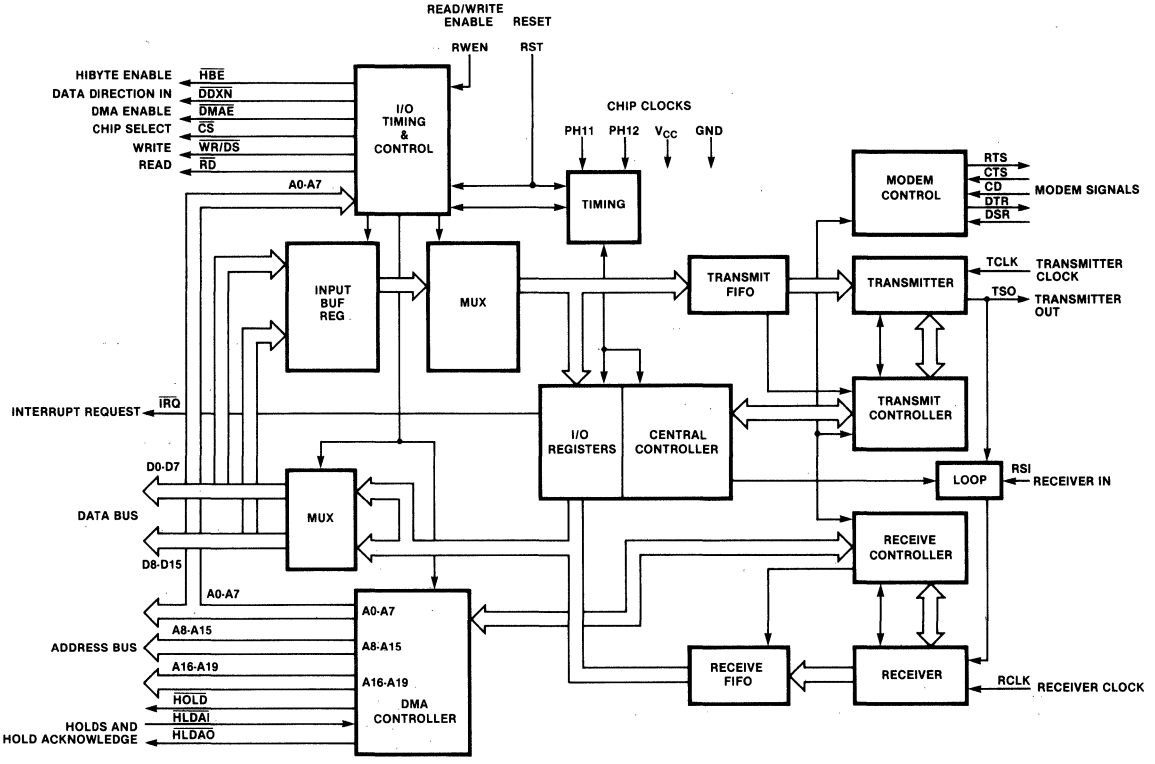


Figure 2 X.25 Interface

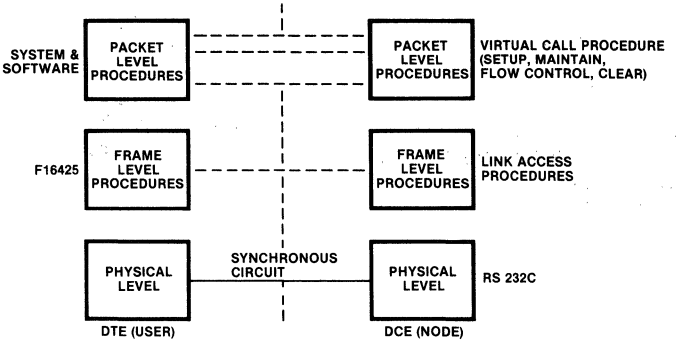


Table 2 F16425 Signal Descriptions

Mnemonic	Name	Description
Chip Select		
CS	Chip Select	An active-low input to enable READ/WRITE of the internal registers during a peripheral access.
Data Bus		
D_0-D_{15}	Data	A 16-bit bidirectional data bus.
Register		
A_0-A_4	Address	Address bits 0-4 are output during DMA access or input during peripheral access; high-impedance output at all other times.
A_5-A_{19}	Address	Address bits 5-19 are output during DMA access; high-impedance output at all other times.
Power		
GND	Ground	
V_{DD}	Power Supply	+5 Volts
Data Direction		
DDIN	Data Direction In	The bidirectional DDIN signal low equals WRITE to FLC: high equals READ from FLC for peripheral access. A DDIN signal low equals WRITE to memory; high equals READ from memory for DMA access.
Interrupt		
IRQ	Interrupt Request	An active-low output indicating that one of the programmed interrupt conditions has occurred.
Transmitter/Receiver		
RCLK	Receive Clock	Direct clock input; the RSI signal changes on the falling edge of the RCLK signal.
RSI	Receive Serial In	A receive serial data input signal.
TCLK	Transmit Clock	Direct clock input, the signal TSO signal changes on the rising edge of the TCLK signal.
TSO	Transmit Serial Out	A transmitted serial output data signal.

Table 1 F16425 Signal Descriptions (Cont'd.)

Mnemonic	Pin No. Name	Description
Modem Control		
CD	Carrier Detect	An input signal
CTS	Clear to Send	An input signal
DSR	Data Set Ready	An input signal
DTR	Data Terminal Ready	An output signal
RTS	Request to Send	An output signal
Chip Control		
DMAE	DMA Enable	A high output when FLC has control of the system buses, and a low output at all other times; can be used to control external 3-state devices.
HBE	High Byte Enable	An active-low input/output signal that enables READ/WRITE to the high-order byte of the data bus; input during peripheral access and output during DMA access. This signal is not used in the 8-bit mode.
HLDAI	Hold Acknowledge In	An active-low input from the CPU or higher priority DMA granting control of the system buses.
HLDAO	Hold Acknowledge Out	An active-low output to a lower priority DMA.
HOLD	Hold	An active-low output requesting control of the system buses.
Φ1	Input Clock	One 0.4- to 10-MHz nonoverlapping two-phase clock input.
RD	Read Strobe	An input/output read strobe if the RWEN input is tied to VDD; not used if RWEN is grounded.
RST	Reset	An active-low input to initialize the internal control and status registers.
RWEN	Read/Write Enable	A low input selects DDIN and DS signals for interfacing M6800-type microprocessors and a high input selects RD and WR signals for interfacing 16,000, 8 D86-type microprocessors.
WR/DS	Write/Data Strobe	A write strobe if the RWEN input is tied to VDD, and a data strobe for READ and WRITE, if the RWEN input is grounded.
WAIT	Wait	Used to extend the memory cycle during a 16-bit read operation.
TEST	Test	A low invokes an internal test sequence.

FAIRCHILD
A Schlumberger Company

F16456
Multiple Protocol Communications Controller

Advance Product Information

Microprocessor Product

Description

The Fairchild F16456 Multiple Protocol Communications Controller (MPCC) is a programmable microprocessor peripheral device that interfaces a computer system to a serial data communication channel with minimum system overhead. It can be used in computer-to-computer or computer-to-terminal communications, or in control of network trunk lines with the F16000 family and other microprocessors.

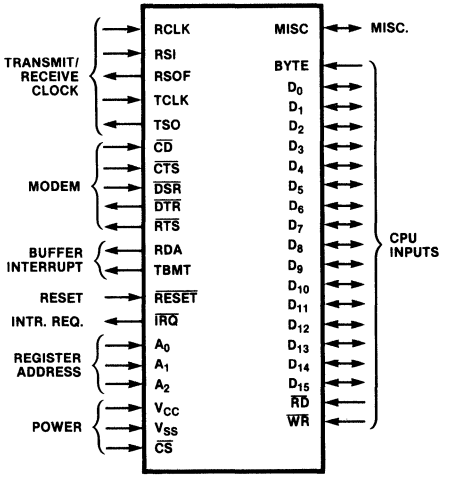
- **F16000 Family, 8080, 8086 Bus Compatible**
- **Satisfies Interface Requirements for Asynchronous Mode and for Synchronous Bit-Oriented Protocol (BOP) or Synchronous Byte-Control Protocol (BCP)**
- **Generates and Tests Error Detection Codes; Generates and Detects Special Characters**
- **Bidirectional Three-State Data Bus Interface (Selectable as 8-Bit or 16-Bit Bus)**
- **Full- or Half-Duplex Operation**
- **Modem Handshake Signals**
- **5- to 8-Bit Character Lengths**
- **Primary or Secondary Station Operation**
- **Normal and Transparent Text Modes**
- **Directly Addressable Registers**
- **Serial Data from dc to 2.0M bps**
- **NRZ or NRZI (Complemented on Zero) Serial Data**
- **Self-Tested Loop-Back Mode**
- **Maintains Data Transparency Through Automatic Manipulation of the Data Stream**
- **TTL-Compatible**
- **Single +5 V Power Supply**
- **40-Pin Plastic or Ceramic Dual-in-Line Package**

The F16456 MPCC is functionally divided into a serial data receiver, serial data transmitter, addressable registers, and data bus control logic. The receiver and transmitter operate at independent rates determined by their clocks. The eight registers contained in the MPCC are directly addressable when using an 8-bit data bus; they are addressed in pairs when a 16-bit data bus is used. The MPCC is manufactured using high-speed XMOS technology. Figure 1 is a block diagram of the F16456.

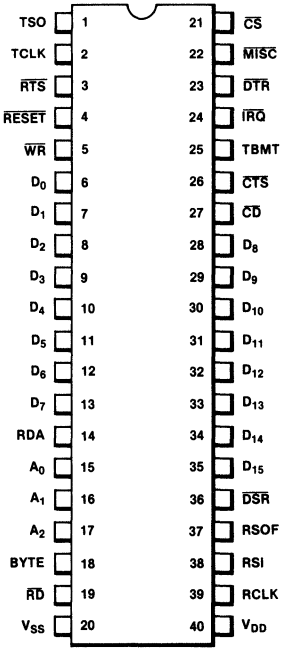
Signal Descriptions

Table 1 lists the input/output signals for the F16456.

Signal Functions



Connection Diagram



7

Figure 1 F16456 Block Diagram

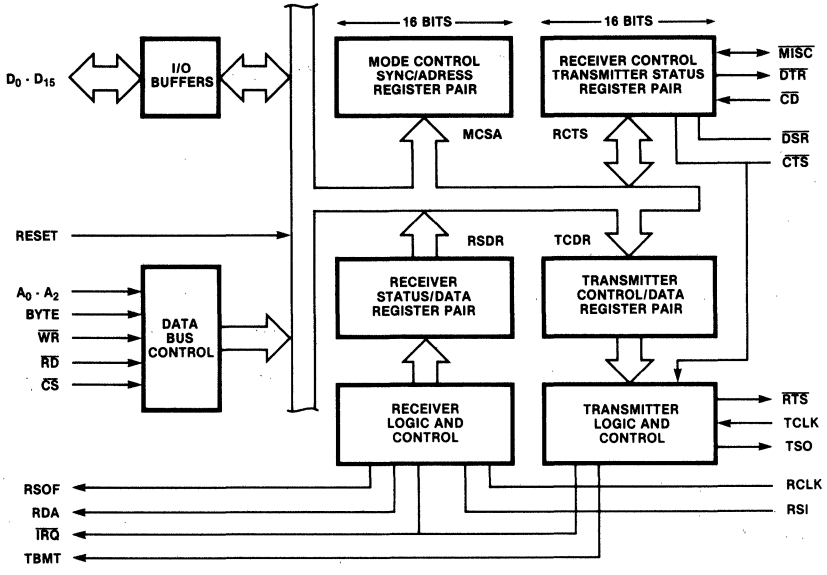


Table 1 F16456 Signal Descriptions

Mnemonic	Pin No.	Name	Description
Power			
V_{CC}	40	Power Supply Input	+5 V power supply
V_{SS}	20	Ground	0 V reference
Misc.			
$\overline{MISC}$	22	Miscellaneous	The DEND bit of the receiver control register (RCR) determines whether the $\overline{MISC}$ pin is an input or an output. When used as an input, a low level sets the end-of-message (EOM) bit in the transmitter control register. When used as an output, the $\overline{MISC}$ pin is general-purpose in nature and is controlled by bit 14 of the RCR.

Table 1 F16456 Signal Descriptions (Cont'd.)

Mnemonic	Pin No.	Name	Description
CPU Inputs			
BYTE	18	Byte	A high-level input that indicates an 8-bit data bus; a low level indicates a 16-bit bus.
D_0-D_{15}	6-13,28-35	Data Bus	A bidirectional 16-bit data bus. An 8-bit data bus interface is obtained by connection D_0 through D_7 to D_8 through D_{15}, respectively, and connecting pin 18 ($\overline{BYTE}$) to +5 V.
$\overline{RD}$	19	Read Pulse	A low-level input that transfers the contents of the addressed register to the data bus if $\overline{CS}$ is low.
$\overline{WR}$	5	Write Pulse	A low-level input that transfers the data bus information to the addressed register is $\overline{CS}$ is low.
RDA	14	Receiver Data Available	A high-level output indicating that an assembled character is in the receiver data buffer; reset on the positive edge of RD when reading from the transmitter buffer.
TBMT	25	Transmitter Buffer Empty	A high-level output indicating that the device is ready to accept another data character from the CPU; reset on the positive edge of $\overline{WR}$ when written to the transmitter buffer.
Reset			
$\overline{RESET}$	4	Reset	A low-level input that disables the transmitter and receiver and initializes the internal control registers and timing.
Interrupt			
$\overline{IRQ}$	24	Interrupt Request	An output signal that goes low to indicate a change in the internal status of the device. The status bits linked to the $\overline{IRQ}$ output are receiver overrun, received end-of-message and async framing error, received parity error, and transmitter underrun. The $\overline{IRQ}$ signal is reset on the trailing edge of $\overline{RD}$ when the associated status register is read.
Register			
A0 A1 A2	15 16 17	Register Address	Input signals that select one of eight 8-bit addressable registers if the BYTE signal is high. If the BYTE signal is low, A0 is not used; A1 and A2 select one of four 16-bit register pairs.

Table 1 F16456 Signal Descriptions (Cont'd.)

Mnemonic	Pin No.	Name	Description
Transmit/Receive Clock			
RCLK	39	Receiver Clock	Timing input signal for the receiver logic.
RSI	38	Received Serial Input	An input signal comprising the received serial data.
RSOF	37	Received Sync or Flag	An output signal that is high for one receiver clock period each time a flag or sync character is received.
TCLK	2	Transmitter Clock	An input signal that provides timing for the transmitter logic.
TSO	1	Transmitted Serial Output	An output signal comprising the transmitted serial data.
Modem			
$\overline{CD}$	27	Carrier Detect	A general-purpose input that can be tested by reading the transmitter status register.
$\overline{CTS}$	26	Clear to Send	An input signal used with the $\overline{RTS}$ signal to enable the transmitter; can be tested by reading the transmitter status register.
$\overline{DSR}$	36	Data Set Ready	A general-purpose input that can be tested by reading the transmitter status register.
$\overline{DTR}$	23	Data Terminal Ready	A general-purpose output that can be set low by programming the $\overline{DTR}$ bit of the receiver control register to a logic one.
$\overline{RTS}$	3	Request to Send	An output signal that can be set low by programming the $\overline{RTS}$ bit of the transmitter control register to a logic one.

FAIRCHILD
A Schlumberger Company

F16488
GPIB Controller

Advance Product Information | Microprocessor Product

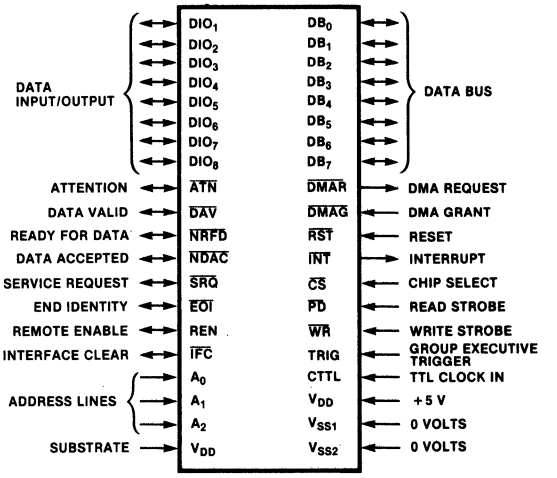
Description

The Fairchild F16488 General-Purpose Interface Bus (GPIB) controller serves as a general IEEE-488 compatible listener/talker and controller within the 16000 microprocessor family for applications in the interconnection of intelligent programmable instruments on the general-purpose interface bus. It is compatible with most MOS microprocessor families.

- **Handles All IEEE 488-1975/78 as Well as 1980 Supplement Functions**
- **Talker, Listener, and Controller Functions**
- **Compliance of All Bus Signals with IEEE 488 and IEC 625 Inputs, Threshold, Termination, and Output Specifications**
- **DMA Access Facilities (Compatible with F16203 and Most Other DMA Controllers)**
- **Single-Phase 10-MHz Maximum TTL Clock**
- **Pass Control Capabilities**
- **Stoppage of Transmission of Unwanted Data Possible Through NBAF Auxiliary Command**
- **On-Board Drivers**
- **Hold-Off on All Data and Hold-Off on Command**
- **End-of-String Recognition by EOS Byte or EOI Pin**
- **Programmable Counter for T1**
- **40-Pin Package**
- **Single 5 V + 10% Power Supply**
- **One-MHz Data Transfer Rate**

Figure 1 is a block diagram of the F16488.

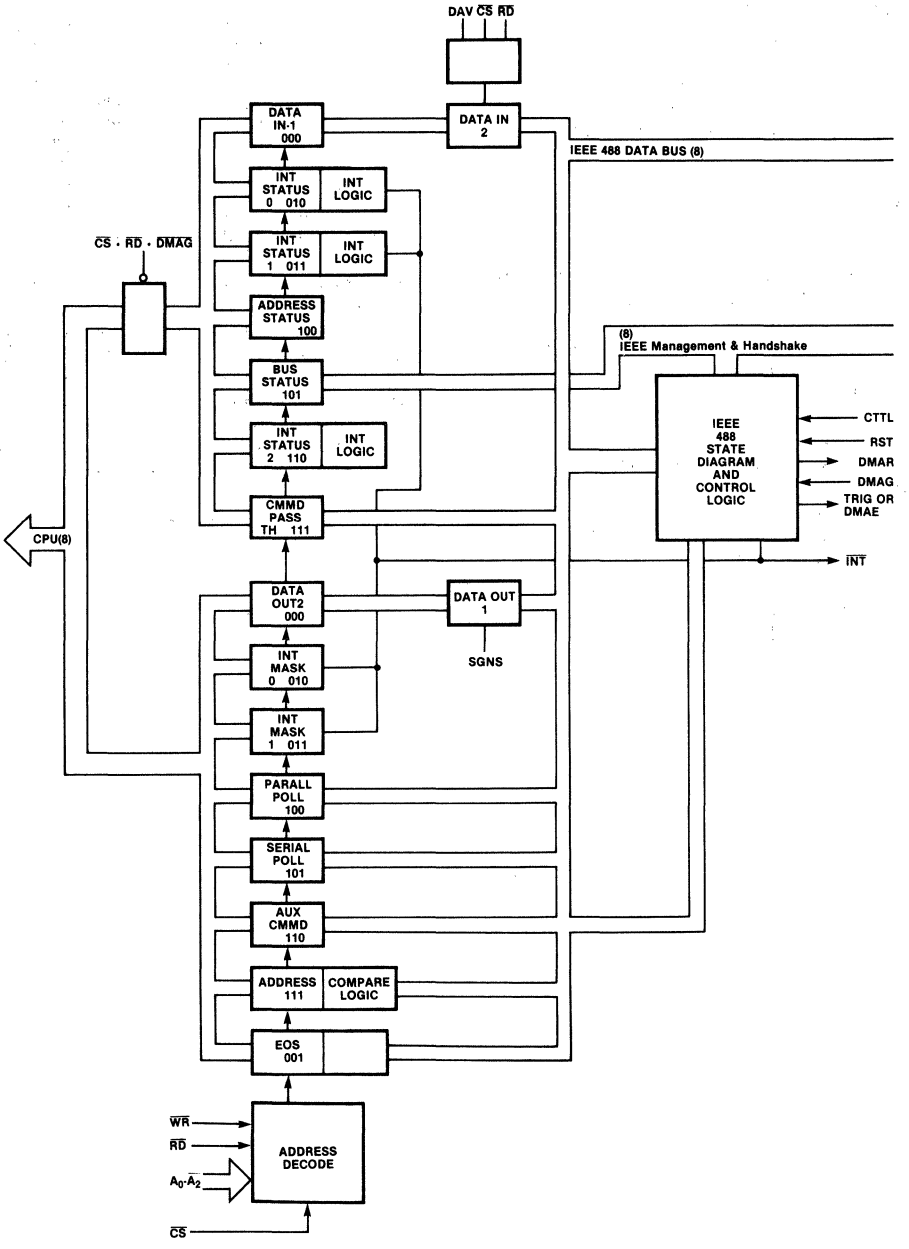
Signal Functions



Note: This is not a pin assignment.

7

Figure 1 F16488 Block Diagram



FAIRCHILD

A Schlumberger Company

F16802 Local Area Network Controller

Advance Product Information

Microprocessor Product

Description

The Fairchild F16802 Local Area Network Controller (LANC) serves as an IEEE-802 token controller within the F16000 microprocessor family. It is intended for application in the interconnection of devices on the token-ring network, and is compatible with most MOS microprocessor families.

- **8-bit and 16-bit CPU Compatibility**
- **Handles IEEE-802 LLC and MAC (token-ring).**

FAIRCHILD
A Schlumberger Company

Section 8
ROM Products

This section contains descriptions of the ROM products. These devices are functionally related to the F6800 microprocessor family and offer varying amounts of read only memory. All of the ROM products are custom chips and are mask-programmable.

FAIRCHILD
A Schlumberger Company

F3532/F68332
F3533
4096 x 8 ROM

Microprocessor Product

Description

The F3532/F68332 and F3533 are 4096 x 8-bit mask-programmable Read Only Memories (ROM), fabricated with n-channel silicon gate technology. They are designed for use in bus-organized systems requiring non-volatile data storage. For ease of use, the F3532/F68332 and F3533 require only a single +5 V power supply, have TTL-compatible inputs and outputs, and, due to their static operation, need no clocking or refreshing.

Electrically identical, the F3532/F68332 and F3533 represent both JEDEC standard pinouts, thus providing compatibility with other available 32K ROMs and EPROMs. To facilitate memory expansion, these devices offer two programmable Chip Select inputs whose active levels are user defined.

The F3532-30/F3533-30 and F3532/F3533-35 are high speed devices, allowing interface with faster generations of NMOS microprocessors.

- **Completely Static Operation**
- **8-Bit Bus Compatible Organization**
- **3-State Outputs**
- **Two Programmable Chip Select Inputs**
- **Single +5 V ± 10% Supply**
- **Fully TTL Compatible**
- **3 Speed Grades—t_{ACC} = 300, 350, 450 ns**
- **Both JEDEC Standard Pinouts**
- **Pin Compatible with Other 32K ROMs and EPROMs**

Pin Names

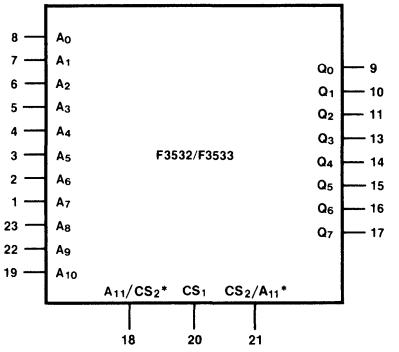
A_0-A_{11}	Address Inputs
CS_1, CS_2	Chip Select Inputs
Q_0-Q_7	Data Outputs

Absolute Maximum Ratings

Voltage on Any Pin Relative to GND	−0.3 V, +7 V
Operating Temperature	0°C, +70°C
Storage Temperature	−65°C, +150°C
Power Dissipation	1 W

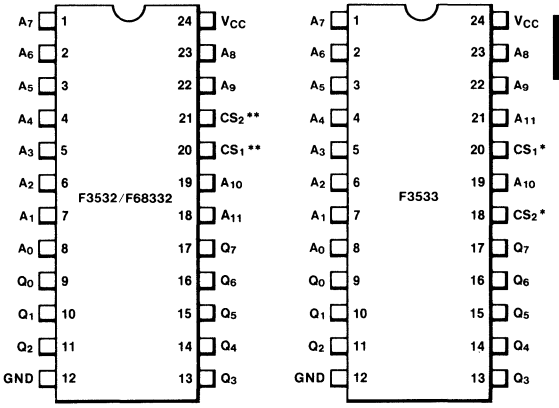
Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operating section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Logic Symbol



V_{CC} = Pin 24
GND = Pin 12
*A_{11} = Pin 18 (F3532/F68332), Pin 21 (F3533);
CS_2 = Pin 21 (F3532/F68332), Pin 18 (F3533)

Connection Diagrams
24-Pin DIP

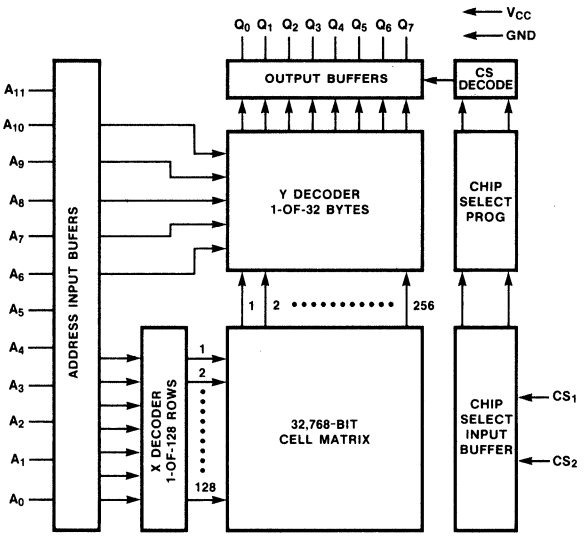


(Top View) (Top View)

**Programmable Chip Selects (see *Custom ROM Programming Information)*

8

Block Diagram



DC Requirements Over operating temperature range

Symbol	Characteristic	Min	Typ	Max	Unit	Note
V_{CC}	Power Supply Voltage	4.5	5.0	5.5	V	
V_{IL}	Input LOW Voltage	−0.3		0.8	V	
V_{IH}	Input HIGH Voltage	2.0		5.5	V	

DC Characteristics Over operating temperature and voltage range

Symbol	Characteristic	Min	Typ	Max	Unit	Note
V_{OL}	Output LOW Voltage			0.4	V	I_{OUT} = 1.6 mA
V_{OH}	Output HIGH Voltage	2.4			V	I_{OUT} = −200 μA
I_{CC}	V_{CC} Power Supply Current			80	mA	1
I_{IN}	Input Leakage Current			2.5	μA	2
I_{OUT}	Output Leakage Current			10	μA	3

Notes on following page.

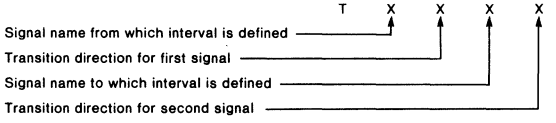
AC Characteristics Over operating temperature and voltage range

IEEE Symbol[6]	Symbol	Characteristic	F35XX-30		F35XX-35		F34XX-45 F68332		Unit	Note
			Min	Max	Min	Max	Min	Max		
TAVAV	t_{CYC}	Cycle Time	300		350		450		ns	
TAVQV	t_{ACC}	Address to Output Access Time		300		350		450	ns	4
TSLQV	t_{CO}	Chip Select to Output Delay Time		120		150		150	ns	4
TSHQZ	t_{DF}	Data Hold After Deselection	10	120	10	150	10	150	ns	4
TAXQZ	t_{DHA}	Data Hold After Address Time	10		10		10		ns	4
	C_{IN}	Input Capacitance		7.5		7.5		7.5	pF	5
	C_{OUT}	Output Capacitance		12.5		12.5		12.5	pF	5

Notes

1. All inputs 5.5 V, $T_A = 0°C$
2. V_{IN} = 0 V to 5.5 V
3. Device unselected: V_{OUT} = 0 V to 5.5 V
4. Measured with 1 TTL Load and 130 pF, transition times = 20 ns
5. Capacitance measured with Boonton Meter
6. Timing Parameter Abbreviations

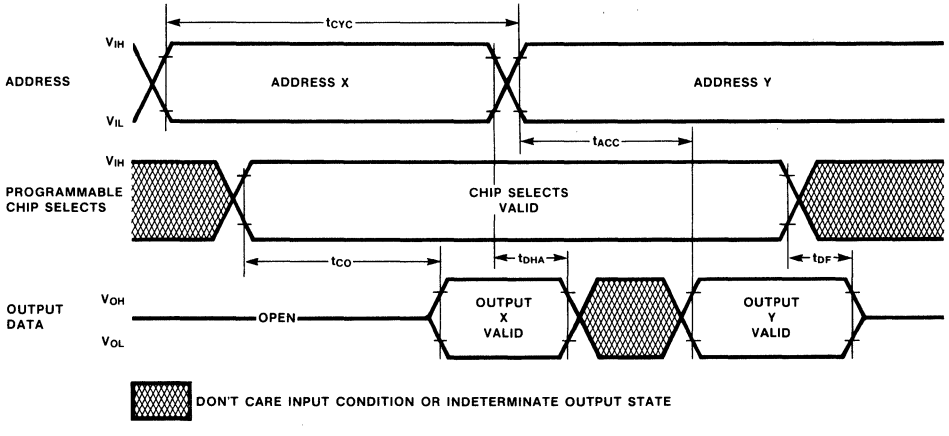
   All timing abbreviations use upper case characters with no subscripts. The initial character is always T and is followed by four descriptors. These characters specify two signal points arranged in a "from-to" sequence that define a timing interval. The two descriptors for each signal point specify the signal name and the signal transitions. Thus the format is:



The signal definitions used in this data sheet are:
A = Address
D = Data In
Q = Data Out
W = Write Enable
E = Chip Enable

The transition definitions used in this data sheet are:
H = transition to HIGH
L = transition to LOW
V = transition to valid
X = transition to invalid or don't care
Z = transition to OFF (high impedance)

Timing Diagram



Custom ROM Programming Information
The customer's unique program code pattern may be submitted to Fairchild in several methods. The most convenient and readily verifiable is in the form of 2708, 2716 or 2732 EPROMs. Program code patterns may also be submitted on Fairchild Formulator MKIII floppy disks or on HP cassette tape in Formulator or MIKBUG* format.

Fairchild Use Only
SL No. _____
Bid Control No. _____
Field Sales Engineer _____
Date Sent _____

Customer Company Name _____
Customer Contact Name _____
Customer Part No. _____

Address _____
Phone No. _____
Fairchild Part No. _____

Customer Input Media
- ☐ 2708 EPROM
- ☐ 2716 EPROM
- ☐ 2732 EPROM
- ☐ Floppy Disk
- ☐ HP Cassette
  - ☐ Formulator Format
  - ☐ MIKBUG Format

Request for Return Media
- ☐ Listing
- ☐ EPROM (include blank EPROMs)

Chip Select Information

	HIGH	LOW	Don't Care
CS_1	☐	☐	☐
CS_2	☐	☐	☐

*MIKBUG is a Motorola trademark.

Formulator Format

0	1	2	3	4	5	6	7	8	9	10	11		M-6	M-5	M-4	M-3	M-2	M-1	M
SOR	L_1	L_0	A_3	A_2	A_1	A_0	T_1	T_0	D_{01}	D_{00}	D_{11}			$D_{(n-1)1}$	$D_{(n-1)0}$	D_{n1}	D_{n0}	CK_1	CK_0

SOR — Start of record defined to be a colon (:)

L_1 L_0 — Length field defined to be the number of packed data bytes per record. Each record is (2*L) + 11 characters in length inclusive of start of record. Length 0 implies end of relocatable module.

A_3 A_2 A_1 A_0 — Address field.

All characters other than SOR are ASCII hexadecimal (0-9, A-F).

T_1 T_0 — Type field.

D_{01} D_{00} . . . $D_{(n)1}$ $D_{(n)0}$ — Data field.

CK_1 CK_0 — Checksum field defined to be negative modulo 256 summation of all bytes since start of record. A summation of all characters in a record, including the checksum, will result in zero.

Ordering Information*

Part No.	Order Code
F3532-30	F353230P, F353230S
F3532-35	F353235P, F353235S
F3532-45	F353245P, F353245S
F3533-30	F353330P, F353330S
F3533-35	F353335P, F353335S
F3533-45	F353345P, F353345S
F68332	F68332P, F68332S

P = Plastic DIP
S = Ceramic DIP
*For extended temperature or military grade, call factory.

FAIRCHILD
A Schlumberger Company

F3564
64K ROM

Advance Product Information

Microprocessor Product

Description

The Fairchild F3564 8192 × 8-bit (64K) mask-programmable read-only memory (ROM) is designed for use in bus-organized systems requiring non-volatile memory storage. Because of its high speed, it readily interfaces with all generatins of NMOS microprocessors.

Fabricated with n-channel silicon-gate technology, the F3564 has industry-standard pinouts and is compatible with other available 24-pin 16K, 32K, and 64K ROMs and EPROMs.

- **Address Latch Feature**
- **Single 5-V Power Supply**
- **Automatic Power-Down**
- **Access Time (t_{AA}) of 250 ns for F3564-25 and 350 ns for F3564-35**
- **Low Power Dissipation (440 mW Maximum Active, 55 mW Maximum Standby)**
- **Fully TTL-Compatible**
- **Three-State Outputs**
- **Mask-Programmable Enable Function**
- **Pin-Compatible with Other Standard 24-Pin 16K, 32K, and 64K ROMs and EPROMs.**

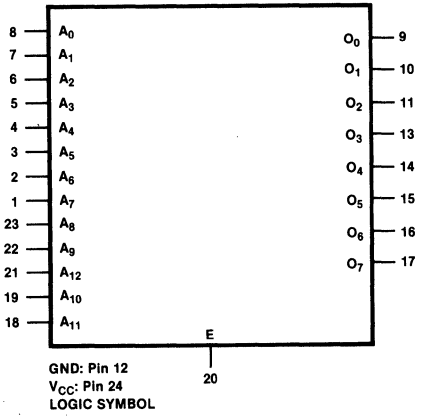
The programmable enable (E) input of the F3564 latches the addresses and controls the active and standby modes of operation (see figure 1); no external latches are required. The active level of the E input and the memory contents are user-defined.

The F3564 requires only a single +5-V power supply, has TTL-compatible inputs and outputs, and, due to its static operation, requires no clocking or refreshing.

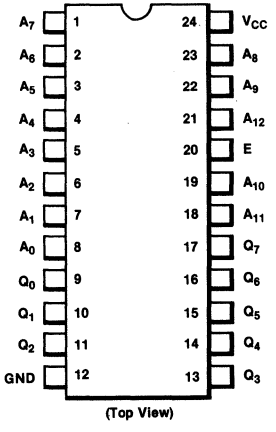
Signal Descriptions

The F3564 signals are described in table 1.

Logic Symbol



Connection Diagram



8

Figure 1 F3564 Block Diagram

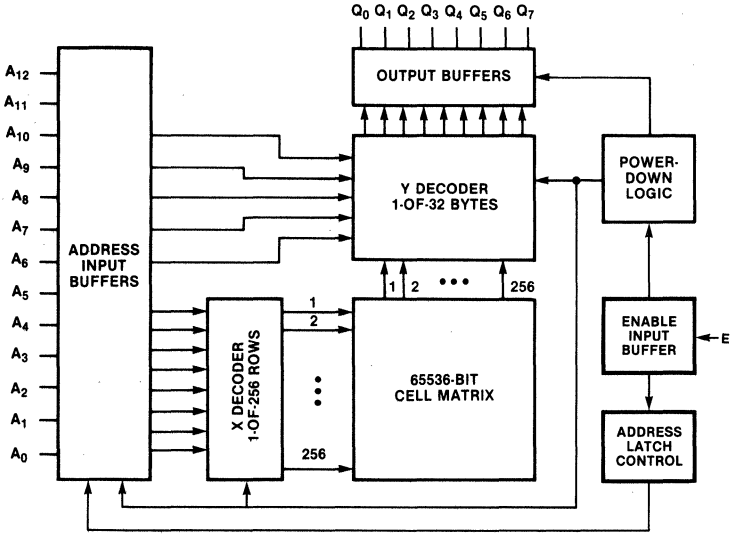


Table 1 Signal Descriptions

Mnemonic	Pin No.	Name	Description
A_0-A_{12}	1-8, 18, 19, 21-23	Address Lines	TTL-compatible input lines that identify the memory location to be read.
E	20	Enable	Programmable input signal that latches the address and controls operating mode. Active level is user-defined.
O_0-O_7	9-11, 13-17	Data Lines	TTL-compatible output lines that contain the data read from the addressed location.
V_{CC}	24	Supply	+5-V power supply
GND	12	Ground	Supply and signal ground

FAIRCHILD
A Schlumberger Company

F3565
64K ROM

Advance Product Information

Microprocessor Product

Description

The Fairchild F3565 8192 × 8-bit (64K) mask-programmable read-only memory (ROM) is designed for use in bus-organized systems requiring non-volatile memory storage. Because of its high speed, it readily interfaces with all generations of NMOS microprocessors.

Fabricated with n-channel silicon-gate technology, the F3565 has industry-standard pinouts and is compatible with other available 24-pin 16K, 32K, and 64K ROMs and EPROMs.

- **Completely Static Operation**
- **Single 5-V Power Supply**
- **Automatic Power-Down**
- **Access Time (t_{AA}) of 250 ns for F3565-25 and 350 ns for F3565-35**
- **Low Power Dissipation (440 mW Maximum Active, 55 mW Maximum Standby)**
- **Fully TTL-Compatible**
- **Three-State Output**
- **Mask-Programmable Enable Function**
- **Pin-Compatible with Other Standard 24-Pin 16K, 32K, and 64K ROMs and EPROMs**

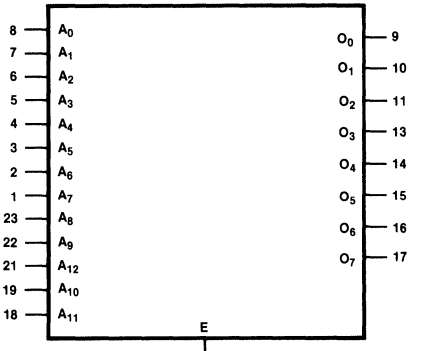
The programmable enable (E) input of the F3565 controls the output and the active/standby modes of operation (see figure 1). The active level of the E input and the memory contents are user-defined.

The F3565 requires only a single +5-V power supply, has TTL-compatible inputs and outputs, and, due to its static operation, requires no clocking or refreshing.

Signal Descriptions

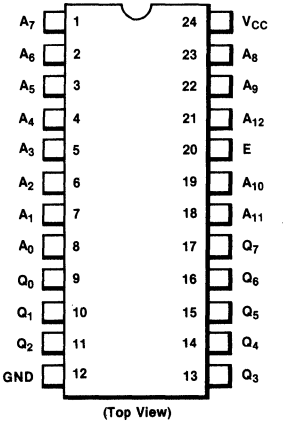
The F3565 signals are described in table 1.

Logic Symbol



LOGIC SYMBOL

Connection Diagram
24-Pin Dip



(Top View)

8

Figure 1 F3565 Block Diagram

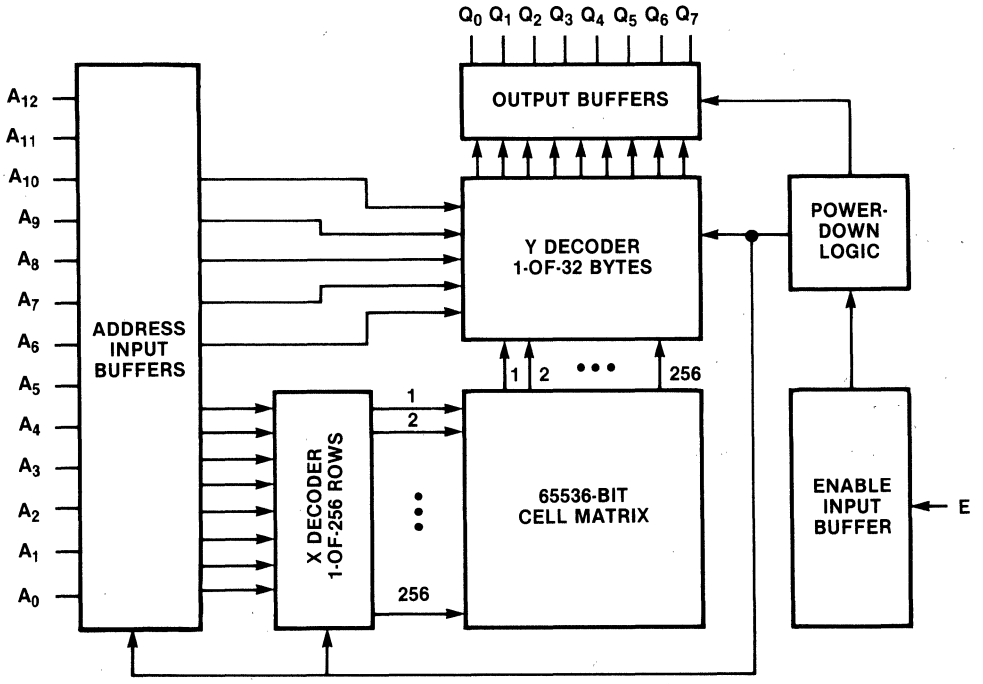


Table 1 Signal Descriptions

Mnemonic	Pin No.	Name	Description
A_0-A_{12}	1-8, 18, 19 21-23	Address Lines	TTL-compatible input lines that identify the memory location to be read.
E	20	Enable	Programmable input signal that latches the address and controls operating mode. Active level is user-defined.
O_0-O_7	9-11, 13-17	Data Lines	TTL-compatible output lines that contain the data read from the addressed location.
V_{CC}	24	Supply	+5-V power supply
GND	12	Ground	Supply and signal ground

FAIRCHILD
A Schlumberger Company

F3566
64K ROM

Advance Product Information | Microprocessor Product

Description

The F3566 8192 × 8-bit (64K) read-only memory (ROM) is designed for use in bus-organized systems requiring non-volatile memory storage. Because of its high speed, it readily interfaces with all generations of NMOS microprocessors.

Fabricated with n-channel silicon-gate technology, the F3566 has industry-standard pinouts and is compatible with other available 24-pin 16K, 32K, and 64K ROMs and EPROMs.

- **Completely Static Operation**
- **Single 5-V Power Supply**
- **High-Speed Data Valid Time of 120 ns**
- **Access Time (t_{AA}) of 250 ns for F3566-25 and 350 ns for F3566-35**
- **Low Power Dissipation (440 mW Maximum Active)**
- **Fully TTL-Compatible**
- **Three-State Output**
- **Mask-Programmable Enable Function**
- **Pin-Compatible with Other Standard 24-Pin 16K, 32K, and 64K ROMs and EPROMs**

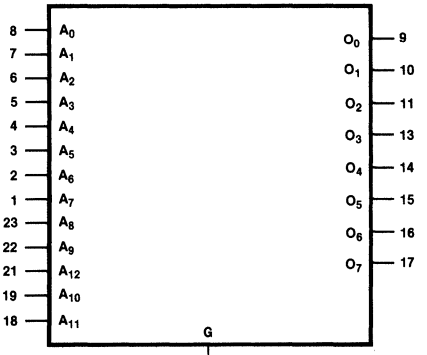
The output enable (G) input controls the output and provides test data and valid time for high-speed microprocessor applications (see figure 1). The G input and the memory contents are user-defined.

The F3566 required only a single +5-V power supply, has TTL-compatible inputs and outputs, and, due to its static operation, needs no clocking or refreshing.

Signal Descriptions

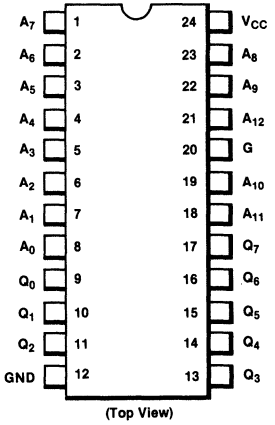
The F3566 signals are described in table 1.

Logic Symbol



LOGIC SYMBOL

Connection Diagram
24-Pin Dip



(Top View)

8

Figure 1 F3566 Block Diagram

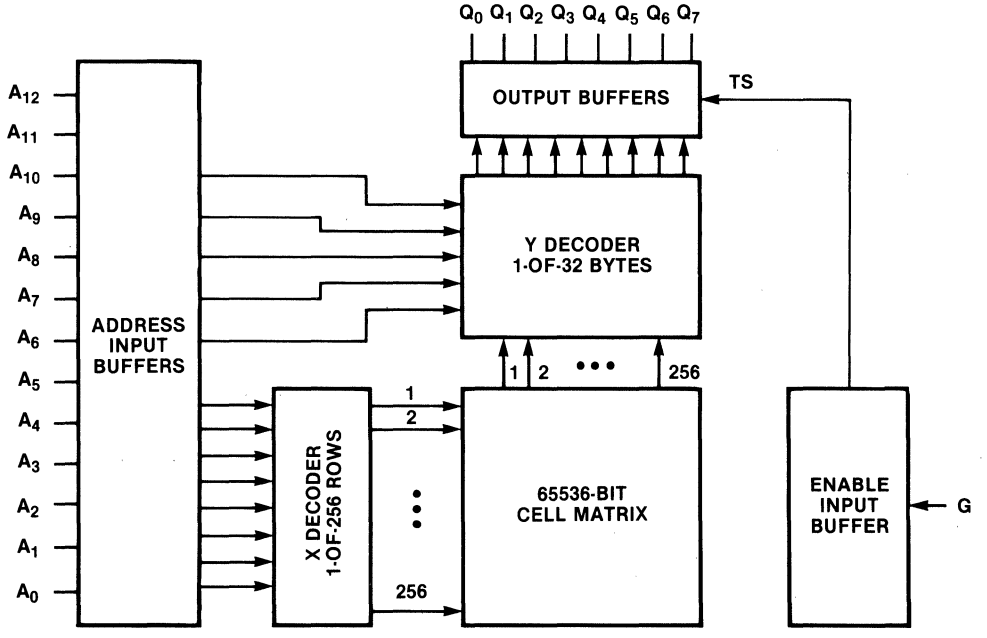


Table 1 Signal Descriptions

Mnemonic	Pin No.	Name	Description
A_0-A_{12}	1-8, 18, 19, 21-23	Address Lines	TTL-compatible input lines that identify the memory location to be read.
G	20	Enable	Programmable input signal that latches the address and controls operating mode. Active level is user-defined.
O_0-O_7	9-11, 13-17	Data Lines	TTL-compatible output lines that contain the data read from the addressed location.
V_{CC}	24	Supply	+5-V power supply
GND	12	Ground	Supply and signal ground

FAIRCHILD
A Schlumberger Company

F3568
64K ROM

Advance Product Information

Microprocessor Product

Description

The Fairchild F3568 8192 × 8-bit (64K) mask-programmable, read-only memory (ROM) is designed for use in bus-organized systems requiring non-volatile memory storage. Because of its high speed, it readily interfaces with all generations of NMOS microprocessors.

Fabricated with n-channel silicon-gate technology, the F3568 has industry standard pinouts and is compatible with other available 28-pin 64K ROMs and EPROMs.

- **Address Latch Feature**
- **Automatic Power-Down**
- **Access Time (t_{AA}) of 250 ns for F3568-25 and 350 ns for F3568-35**
- **Low Power Dissipation (440 mW, Maximum, Active; 55 mW, Maximum, Standby)**
- **Fully TTL-Compatible**
- **Three-State Outputs**
- **Mask-Programmable Enable Function**
- **Single 5 V Power Supply**
- **Pin-Compatible with Other Standard 28-Pin 64K ROMs and EPROMs**

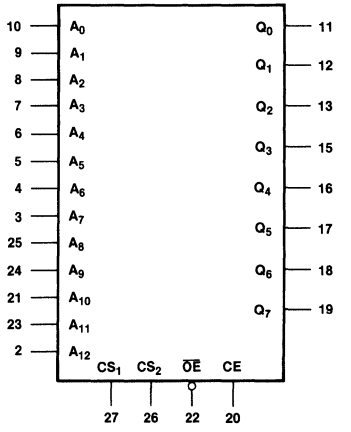
The chip enable (CE) input of the F3568 latches the addresses and controls the active and standby modes of operation; the output enable (OE) input controls the chip outputs and provides fast data valid time for high-speed microcomputer applications (see figure 1). Two chip select (CS) inputs are provided for memory expansion. The active levels of the CE and CS inputs, and the memory contents, are user-defined.

The F3568 requires only a single +5 V power supply, has TTL-compatible inputs and outputs, and, due to its static operation, requires no clocking or refreshing.

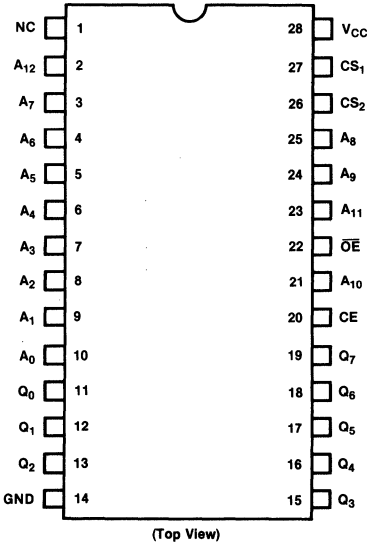
Signal Descriptions

The input/output signal functions of the F3568 are described in table 1.

Logic Symbol



Connection Diagram
28-Pin Dip



(Top View)

8

Figure 1 F3568 Block Diagram

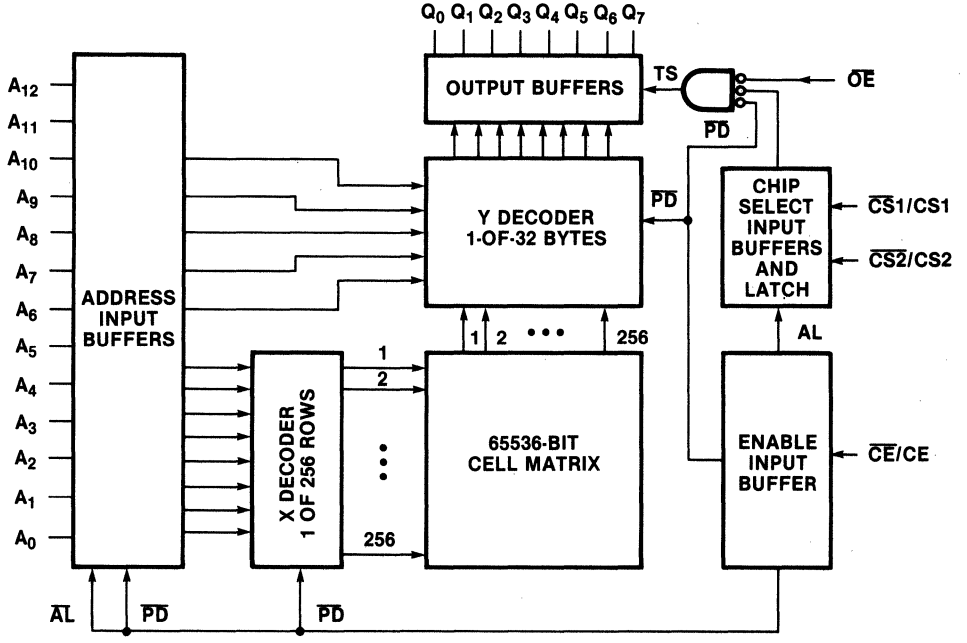


Table 1 Signal Functions

Mnemonic	Pin No.	Name	Description
A_0-A_{12}	2-10, 21, 23-25	Address Lines	TTL-compatible input lines that identify the memory location to be read
CE	20	Chip Enable	Programmable input signal that latches the address and controls operating mode; active level is user-defined.
CS_1, CS_2	26, 27	Chip Select	Programmable input signals that allow memory expansion; active level is user-defined.
GND	14	Ground	Supply and signal ground
OE	22	Output Enable	Input signal that controls outputs and provides fast data valid time
Q_0-Q_7	11-13, 15-19	Data Lines	TTL-compatible output lines that contain the data read from the addressed location
V_{CC}	28	Supply	+5 V power supply

FAIRCHILD
A Schlumberger Company

F3569
64K ROM

Advance Product Information

Microprocessor Product

Description

The Fairchild F3569 8192 × 8-bit (64K) mask-programmable, read-only memory (ROM) is designed for use in bus-organized systems requiring non-volatile memory storage. Because of its high speed, it readily interfaces with all generations of NMOS microprocessors.

Fabricated with n-channel silicon-gate technology, the F3569 has industry-standard pinouts and is compatible with other available 28-pin 64K ROMs and EPROMs.

- **Automatic Power-Down**
- **Access Time (t_{AA}) of 250 ns for F3569-25 and 350 ns for F3569-35**
- **Low Power Dissipation (440 mW, Maximum, Active; 55 mW, Maximum, Standby)**
- **Fully TTL-Compatible**
- **Three-State Outputs**
- **Mask Programmable Enable Function**
- **Single 5 V Power Supply**
- **Completely Static Operation**
- **Pin-Compatible with Other Standard 28-Pin 64K ROMs and PROMs**

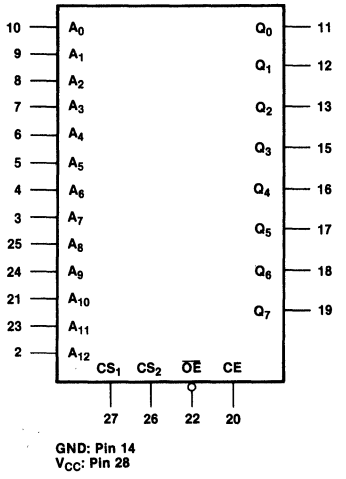
The chip enable (CE) input of the F3569 controls the active and standby modes of operation; the output enable (OE) input controls the chip output and provides fast data available time for high-speed microcomputer applications (see figure 1). Two chip select (CS) inputs are provided for memory expansion. The active levels of the CE and CS inputs, and the memory contents, are user-defined.

The F3569 requires only a single +5 V power supply, has TTL-compatible inputs and outputs, and, due to its static operation, requires no clocking or refreshing.

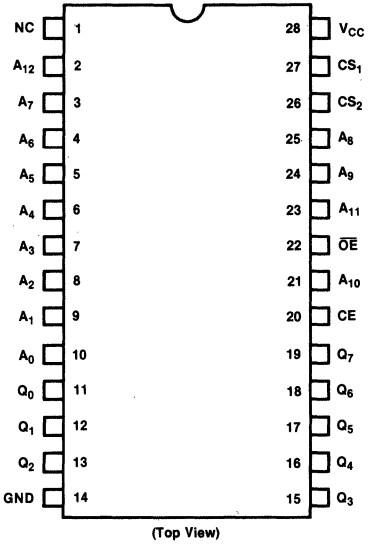
Signal Descriptions

The input/output signal functions of the F3569 are described in table 1.

Logic Symbol



Connection Diagram
28-Pin Dip



(Top View)

8

Figure 1 F3569 Block Diagram

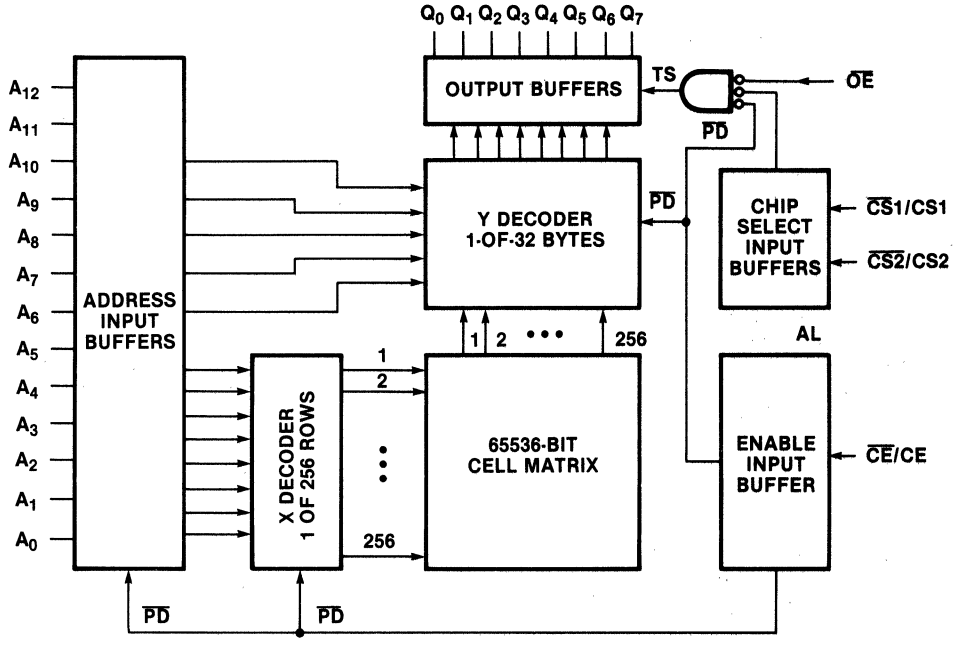


Table 1 Signal Functions

Mnemonic	Pin No.	Name	Description
A_0-A_{12}	2-10, 21, 23-25	Address Lines	TTL-compatible input lines that identify the memory location to be read
CE	20	Chip Enable	Programmable input signal that controls operating mode; active level is user-defined.
CS_1, CS_2	26, 27	Chip Select	Programmable input signals that allow memory expansion; active level is user-defined.
GND	14	Ground	Supply and signal ground
OE	22	Output Enable	Input signal that controls outputs and provides fast data valid time
Q_0-Q_7	11-13, 15-19	Data Lines	TTL-compatible output lines that contain the data read from the addressed location
V_{CC}	28	Supply	+5 V power supply

FAIRCHILD
A Schlumberger Company

F3570
64K ROM

Advance Product Information

Microprocessor Product

Description

The Fairchild F3570 8192 × 8-bit (64K) mask-programmable, read-only memory (ROM) is designed for use in bus-organized systems requiring non-volatile memory storage. Because of its high speed, it readily interfaces with all generations of NMOS microprocessors.

Fabricated with n-channel silicon-gate technology, the F3570 has industry-standard pinouts and is compatible with other available 28-pin 64K ROMs and EPROMs.

- **Access Time (t_{AA}) of 250 ns for F3570-25 and 350 ns for F3570-35**
- **High-Speed Data Valid Time of 120 ns**
- **Low Power Dissipation (440 mW, Maximum, Active)**
- **Fully TTL-Compatible**
- **Three-State Outputs**
- **Mask-Programmable Chip Select Active Levels**
- **Single 5 V Power Supply**
- **Completely Static Operation**
- **Pin-Compatible with Other Standard 28-Pin 64K ROMs and EPROMs**

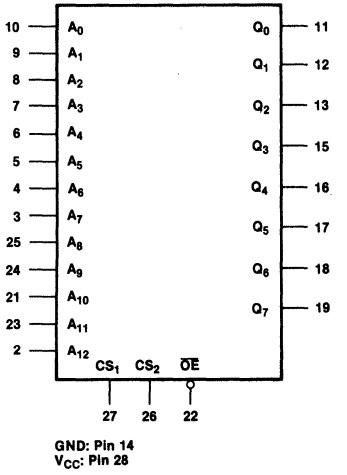
The output enable (OE) input controls the chip output and provides fast data valid time for high-speed microprocessor applications (see figure 1). Two chip select (CS) inputs are provided for memory expansion. The active levels of the CS inputs, and the memory contents, are user-defined.

The F3570 requires only a single +5 V power supply, has TTL-compatible inputs and outputs, and, due to its static operation, requires no clocking or refreshing.

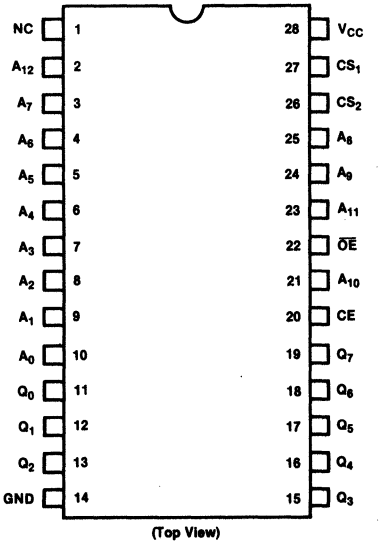
Signal Descriptions

The input/output signal functions of the F3570 are described in table 1.

Logic Symbol



Connection Diagram
28-Pin Dip



(Top View)

8

Figure 1 F3570 Block Diagram

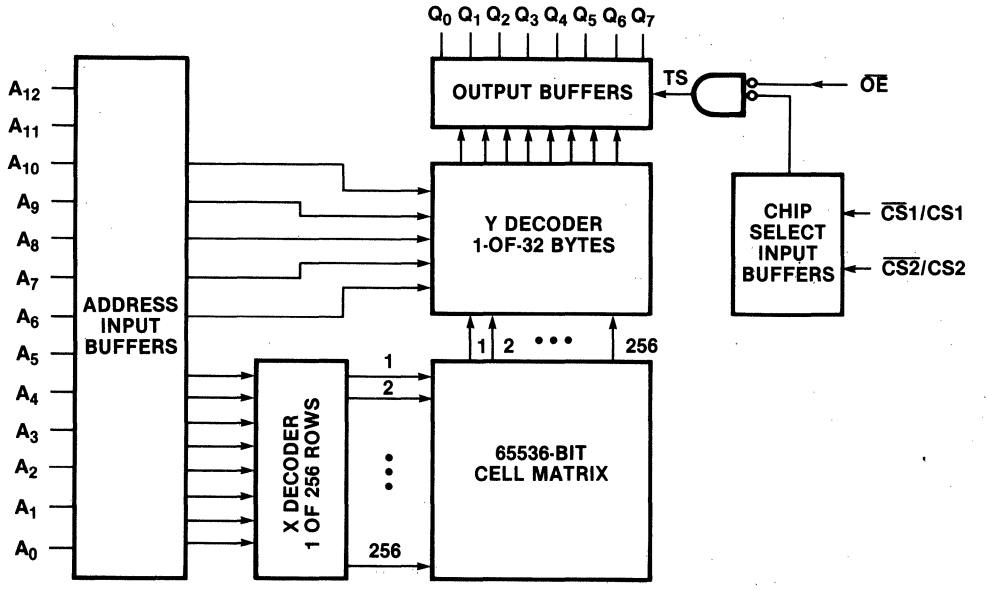


Table 1 Signal Functions

Mnemonic	Pin No.	Name	Description
A_0-A_{12}	2-10, 21, 23-25	Address Lines	TTL-compatible input lines that identify memory location to be read
CS_1, CS_2	26, 27	Chip Select	Programmable input signals that allow memory expansion; active level is user-defined.
GND	14	Ground	Supply and signal ground
OE	22	Output Enable	Input signal that controls outputs and provides fast data valid time
Q_0-Q_7	11-13, 15-19	Data Lines	TTL-compatible output lines that contain the data read from the addressed location
V_{CC}	28	Supply	+5 V power supply

FAIRCHILD
A Schlumberger Company

F35316/F68316
2048 x 8 ROM

Microprocessor Products

Description

The F35316/F68316 is a mask-programmable byte-organized MOS Read Only Memory (ROM) designed for use in bus-organized systems requiring non-volatile data storage. It is fabricated with n-channel silicon-gate technology. For ease of use, the F35316/F68316 operates from a single +5 V power supply, inputs and outputs are TTL and DTL compatible, and the device needs no clocks or refreshing because of its static operation.

The F35316/F68316 is compatible with the F6800, F8 and other microcomputer families providing read only storage in byte increments. To facilitate memory expansion, the device contains three programmable Chip Select inputs providing any combination of active HIGH or LOW or an optional DON'T CARE state coupled with output wired-OR capability. Chip select code and memory content are user defined and are fixed during the masking process.

The F35316/F68316 provides maximum circuit density, reliability and performance yet maintains low power dissipation and yields significant cost advantages over an EPROM approach.

- **2048 x 8-BIT BUS-COMPATIBLE ORGANIZATION**
- **FULLY STATIC OPERATION**
- **3-STATE DATA OUTPUTS FOR WIRED-OR CAPABILITY**
- **MASK-PROGRAMMABLE CHIP SELECTS FOR SIMPLIFIED MEMORY EXPANSION**
- **SINGLE +5 V ± 10% POWER SUPPLY**
- **TTL AND DTL-COMPATIBLE INPUTS**
- **MULTIPLE SPEED GRADES**
  t_{ACC} = 250 ns, 300 ns (F35316)
  t_{ACC} = 350 ns, 450 ns, 500 ns (F68316)
- **DIRECTLY COMPATIBLE WITH 2316E**
- **PIN COMPATIBLE WITH F2708 AND F2716 EPROMs**

Pin Names

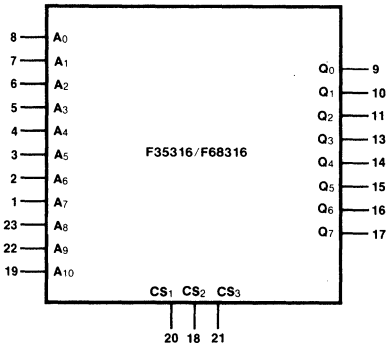
A_0-A_{10}	Address Inputs
CS_1-CS_3	Chip Select Inputs
Q_0-Q_7	Data Outputs

Absolute Maximum Ratings

Voltage on Any Pin Relative to GND	−0.3 V to +7 V
Operating Temperature	0°C to +70°C
Storage Temperature	−65°C to +150°C
Power Dissipation	1 W

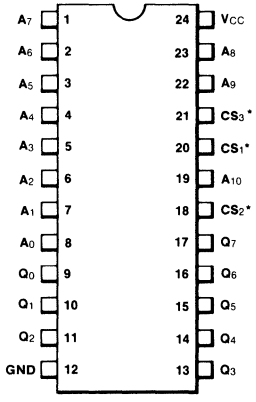
Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operating section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Logic Symbol



V_{CC} = Pin 24
GND = Pin 12

Connection Diagram
24-Pin DIP

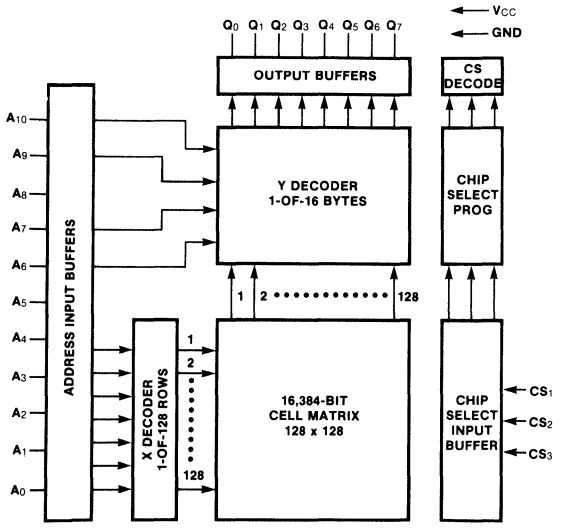


(Top View)

*Programmable Chip Selects

8

Block Diagram



DC Requirements Over operating temperature range

Symbol	Characteristic	Min	Typ	Max	Unit
V_{CC}	Power Supply Voltage	4.75	5.0	5.25	V
V_{IL}	Input LOW Voltage	−0.5		0.8	V
V_{IH}	Input HIGH Voltage	2.0		5.5	V

DC Characteristics Over operating temperature and voltage range

Symbol	Characteristic	Min	Typ	Max	Unit	Notes
I_{CC}	V_{CC} Power Supply Current			110	mA	1
I_{IN}	Input Leakage Current			2.5	μA	2
I_{OUT}	Output Leakage Current			10	μA	3
V_{OL}	Output LOW Voltage			0.4	V	I_{OUT} = 1.6 mA
V_{OH}	Output HIGH Voltage	2.4			V	I_{OUT} = −200 μA

Notes on following page.

AC Characteristics (F35316) Over operating temperature and voltage range

IEEE Symbol[6]	Symbol	Characteristic	F35316-25		F35316-30		Unit	Note
			Min	Max	Min	Max		
TAVAV	t_{CYC}	Cycle Time	250		300		ns	
TAVQV	t_{ACC}	Address to Output Delay Time		250		300	ns	4
TSLQV	t_{CO}	Chip Select to Output Delay Time		150		150	ns	4
TSHQZ	t_{DF}	Data Hold After Deselection	10	150	10	150	ns	4
TAXQZ	t_{DHA}	Data Hold After Address Time	10		10		ns	4
	C_{IN}	Input Capacitance		7.5		7.5	pF	5
	C_{OUT}	Output Capacitance		12.5		12.5	pF	5

AC Characteristics (F68316) Over operating temperature and voltage range

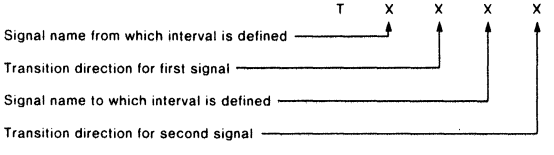
IEEE Symbol[6]	Symbol	Characteristic	F68316-35		F68316-45		Unit	Note
			Min	Max	Min	Max		
TAVAV	t_{CYC}	Cycle Time	350		450		ns	
TAVQV	t_{ACC}	Address to Output Delay Time		350		450	ns	4
TSLQV	t_{CO}	Chip Select to Output Delay Time		150		150	ns	4
TSHQZ	t_{DF}	Data Hold After Deselection	10	150	10	150	ns	4
TAXQZ	t_{DHA}	Data Hold After Address Time	10		10		ns	4
	C_{IN}	Input Capacitance		7.5		7.5	pF	5
	C_{OUT}	Output Capacitance		12.5		12.5	pF	5

8

Notes

1. All inputs 5.5 V, T_A = 0°C
2. V_{IN} = 0 V to 5.5 V
3. Device unselected: V_{OUT} = 0 V to 5.5 V
4. Measured with 1 TTL load and 130 pF, transition times = 20 ns
5. Capacitance measured with Boonton Meter
6. Timing Parameter Abbreviations

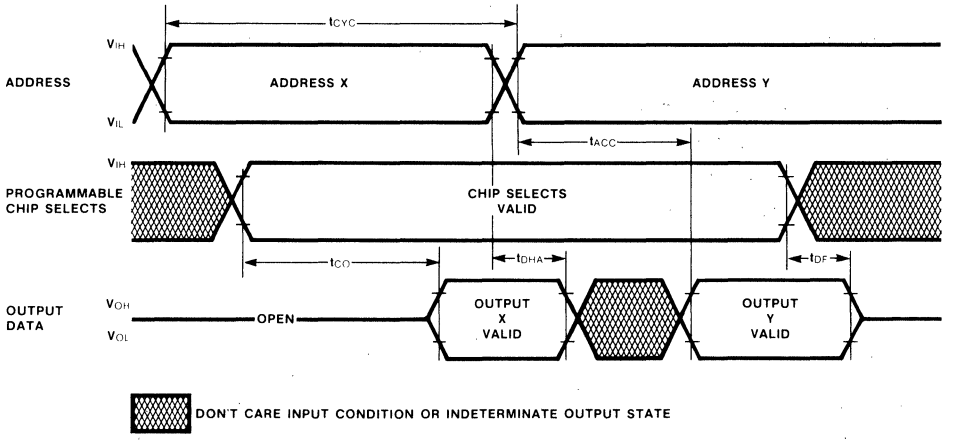
   All timing abbreviations use upper case characters with no subscripts. The initial character is always T and is followed by four descriptors. These characters specify two signal points arranged in a "from-to" sequence that define a timing interval. The two descriptors for each signal point specify the signal name and the signal transitions. Thus the format is:



The signal definitions used in this data sheet are:
A = Address
D = Data In
Q = Data Out
W = Write Enable
E = Chip Enable

The transition definitions used in this data sheet are:
H = transition to HIGH
L = transition to LOW
V = transition to valid
X = transition to invalid or don't care
Z = transition to OFF (high impedance)

Timing Diagram



Custom ROM Programming Information

The customer's unique program code pattern may be submitted to Fairchild in several methods. The most convenient and readily verifiable is in the form of 2708, 2716 or 2732 EPROMs. Program code patterns may also be submitted on Fairchild Formulator MKIII floppy disks or on HP cassette tape in Formulator or MIKBUG* format.

Fairchild Use Only

SL No. _____
Bid Control No. _____
Field Sales Engineer _____
Date Sent _____

Customer Company Name _____
Customer Contact Name _____
Customer Part No. _____

Address _____
Phone No. _____
Fairchild Part No. _____

Customer Input Media

- ☐ 2708 EPROM
- ☐ 2716 EPROM
- ☐ 2732 EPROM
- ☐ Floppy Disk
- ☐ HP Cassette
  - ☐ Formulator Format
  - ☐ MIKBUG Format

Request for Return Media

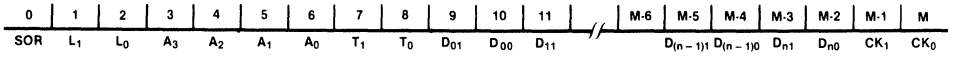
- ☐ Listing
- ☐ EPROM (include blank EPROMs)

Chip Select Information

	HIGH	LOW	Don't Care
CS_1	☐	☐	☐
CS_2	☐	☐	☐
CS_3	☐	☐	☐

*MIKBUG is a Motorola trademark.

Formulator Format



SOR — Start of record defined to be a colon (:)

L_1 L_0 — Length field defined to be the number of packed data bytes per record. Each record is (2*L) + 11 characters in length inclusive of start of record. Length 0 implies end of relocatable module.

A_3 A_2 A_1 A_0 — Address field.

T_1 T_0 — Type field.

D_{01} D_{00} . . . $D_{(n)1}$ $D_{(n)0}$ — Data field.

CK_1 CK_0 — Checksum field defined to be negative modulo 256 summation of all bytes since start of record. A summation of all characters in a record, including the checksum, will result in zero.

All characters other than SOR are ASCII hexadecimal (0-9, A-F).

Ordering Information*

Part No.	Order Code
F35316-25	F3531625P, F3531625S
F35316-30	F3531630P, F3531630S
F68316-35	F3531635P, F3531635S
F68316-45	F3531645P, F3531645S

P = Plastic DIP
S = Ceramic DIP
* For extended temperature or military range, call factory.

8

FAIRCHILD
A Schlumberger Company

Section 9
Development Systems and Software

General

The following is data that describes the design aids available for hardware and software development and emulation in the creation of Fairchild microprocessor-based systems.

FAIRCHILD
A Schlumberger Company

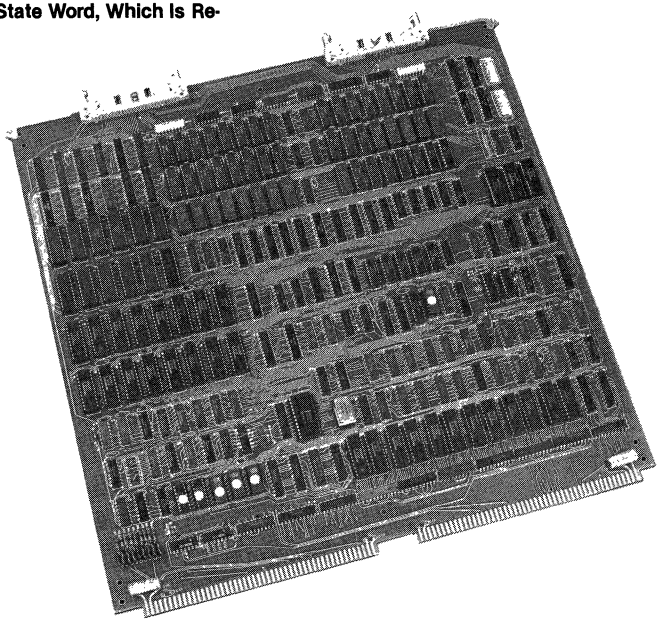
EMUTRAC™
Emulation and Tracing System

Advance Product Information | Microprocessor Product

Description

The Fairchild EMUTRAC is a powerful, cost-effective, in-circuit emulation and tracing system that supports microcomputer system development. The EMUTRAC system allows simultaneous and interactive hardware and software development, which permits control, interrogation, revision, and debugging of a microcomputer system in its own real-time environment. Software may be developed and debugged with or without complete prototype hardware.

- **Single Controller Fits Within the FS-I Chassis.**
- **Interchangeable External Modules Individually Support F3870, F6800, F6809, and F9445 Microprocessors.**
- **EMUTRAC Provides Optional Substitution for CPU and I/O Peripherals, as Well as Memory, in Prototype Systems.**
- **Address-Steering Allows Selective Substitution for Prototype System Memory, in Blocks of 64 Words.**
- **8K Words (or 16K Bytes) of Mappable Substitution-RAM is Provided, Using 2114 or 2148 Devices.**
- **4-Bit Tags, Which Aid in Breakpoint-Marking, Can Be Associated with Individual Substitution-Memory Locations, 256-Location Blocks of Prototype Memory, or I/O Device Accesses.**
- **A Tag, User-Assignable Probes, Bus Data, and Functions of Key Microprocessor and EMUTRAC-Internal Signals Comprise the 48-Bit Machine-State Word, Which Is Re-evaluated for Each Bus Cycle.**
- **The Breakpoint Comparator Examines the 48-Bit Machine-State Word During Each Bus Cycle, and Can Detect Eight Simultaneous Breakpoint Conditions.**
- **A Programmable Micro-Sequencer Responds During Each Cycle to the Detected Condition by:**
  - **— Conditional Change in Sequence, with "Jump" or "Step" Functions;**
  - **— Conditional Update of Two Independent Delay-Counters;**
  - **— Optional Issue of Four Independent Pulses, to Sync External Tests;**
  - **— Conditional Recording of One Trace-Frame;**
  - **— Optional "Pause," "Interrupt," etc., Functions.**
- **Each Trace-Frame Word Captures 64 Bits, Composed of the Address Issued During Bus-Cycle and Machine-State Word.**
- **Interactive Hardware/Software Debugging is Simplified, with Symbolic Location/Variable-Names, Instruction Mnemonics, and Signal-Names.**
- **Simple, English-like Commands Control the Emulation Process.**
- **Command Language Provides REPEAT, File-INCLUDE, and MACRO Capabilities, as Well as Session-Logfile and Selective-Printout Generation.**



9

™EMUTRAC is a trademark of Fairchild Camera and Instrument Corporation.

System Function

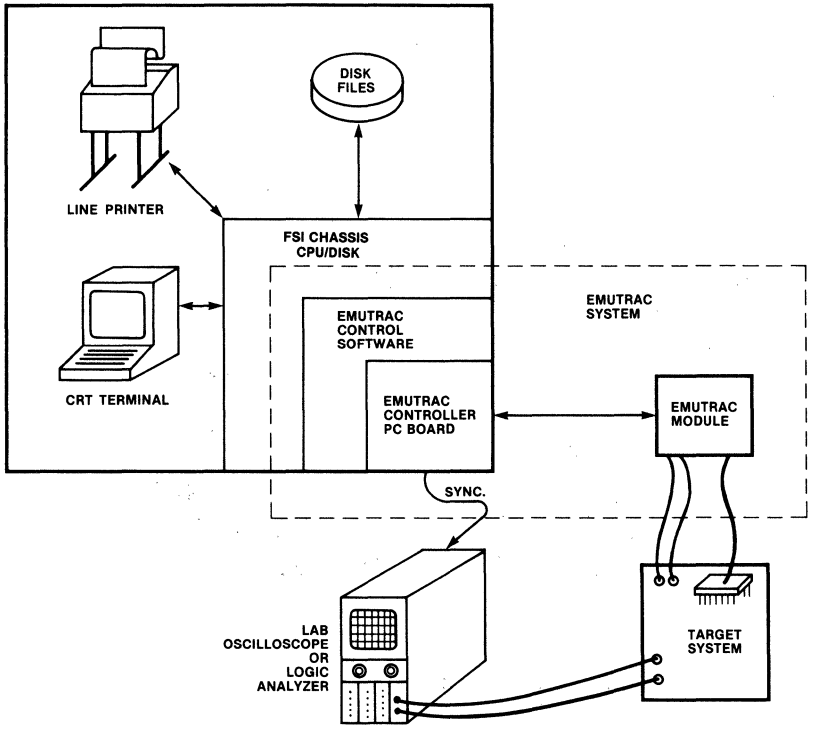
The EMUTRAC system combines the functions of console operations, a symbolic debugger, a logic analyzer, and a substitution CPU and memory. The system consists of an EMUTRAC controller card that plugs into the FS-I development system and an external module that interfaces the controller to the target system (see figure 1).

The EMUTRAC system supports all Fairchild microprocessor families, with processor-independent logic on the controller card and logic unique to a specific processor residing in the external module. Different microprocessors can be supported using different EMUTRAC modules with the same EMUTRAC controller card.

Operator Interface

The EMUTRAC control software, which runs on the FS-I, has simple setup commands that provide explicit control of the memory mapping, tag attachment, breakpoint definition, and sequencer action functions. Additional commands provide the block load/dump functions for RAM definition, as well as the interactive examine/deposit-search operations usually provided in a debugger. Simple commands provide start, stop, and single-cycle control for the emulation; other commands control the operator interface providing log file and print generation, checkpoint creation and retrieval, and REPEAT, INCLUDE, and MACRO commands. The control software provides an easy-to-use, concise command structure, with HELP commands to aid on-line learning, yet aids the accomplished user through command files to perform repetitious tasks.

Figure 1 EMUTRAC System



Memory Substitution and Initialization
The EMUTRAC system's RAM can selectively substitute for sections of the prototype system's memory. Thus, table-modifications or code-patches can be made to RAM, and the results verified, without time-consuming PROM-programming or ROM-masking. Similarly, the RAM can be used as memory-expansion for the prototype system, permitting extra-large programs (with diagnostic or debugging aids) to be used during development and test.

Block-initialization of EMUTRAC RAM or prototype RAM (or of any control-RAM within the EMUTRAC) can be easily accomplished with the "load <ramname> from <filename>" command. The complementary "dump <ramname> into <filename>" provides a simple way to capture the current contents of any memory. Together, these commands allow "snapshots" to be taken, for later comparison and analysis or for quick state-restoration between test runs.

Operator Control
The console functions of the EMUTRAC system consist of four groups: run control, memory examine and deposit, I/O register examine and deposit, and CPU register examine and deposit. The run control comprises STOP, RESET, START, CONTINUE, and single-instruction STEP commands. The memory, I/O register, and CPU register examine and deposit controls allow the user to inspect and modify the state of the microprocessor, I/O device, and system memory registers. Locations examined can be displayed in symbolic form, and modifications can be made in terms of user-defined symbols and mnemonic instruction codes.

Program Breakpoints
The breakpoint feature provides controlled interruptions or normal program flow when the user-selected pattern of status conditions exists, so that memory, registers, and CPU status can be interrogated and traced. To aid detection of ranges or scattered instances of address- or I/O-access, 4-bit-per-location tags are provided in EMUTRAC memory; thus, improper memory WRITE operations, access to non-existent memory or I/O devices, and references to key variables are all simple to identify. The EMUTRAC breakpointing facility is extremely powerful; up to eight independent breakpoints can be simultaneously monitored.

Real-Time Trace Control
The tracing feature of the EMUTRAC system functions like a dedicated logic analyzer, giving the user a record of up to 255 previous events. The "audit trail" thus created can be used to find the cause of system failure. The EMUTRAC system, however, offers much more than a normal logic analyzer.

Trace-frame generation is controlled by the programmable sequencer. Detection of a breakpoint can trigger capture of consecutive machine cycles, and counter controls can "center" this capture window as desired. Additionally, the trace log can be considerably filtered to include only those events surrounding the trigger that satisfy additional conditions, thereby making better use of available trace memory. Alternatively, short packets of trace information can be recorded in response to multiple trigger conditions encountered during testing.

Software Timing
The software timing feature, which works under the control of the breakpoint sequencer, allows the user to acquire statistics on the performance of the microprocessor system software modules. The timer allows the user to measure the execution time of a block of code, as well as the number of times that block of code was used during the execution of a given program. This permits the user to estimate the performance of the total system and provides direction for optimization efforts. It can also be used to identify failing sequences that take significantly smaller or larger amounts of time than anticipated.

Command-Language Features
Commands to the EMUTRAC system are issued as a sequence of simple, English-like sentences; diagnostic messages in response to command errors, and the HELP command assist new users in operating the system. The accomplished user is assisted by language features such as:

IF <expression> (. . .) ELSE (. . .)

which allows conditional command-issue,

REPEAT <count> (. . .)

which reissues a set of commands several times,

INCLUDE <filename>

which issues a pre-recorded sequence of commands, and

MACRO <arglist> (. . .)

which constructs sequences with replaceable elements.

During the emulation, all commands are recorded in a session-log file as they are issued; this file could, for instance, be printed as documentation of test results. The run can later be duplicated, or extended, by simply issuing the saved log-data as commands with an INCLUDE statement.

FAIRCHILD
A Schlumberger Company

F38E70
Programming Board

Microprocessor Product

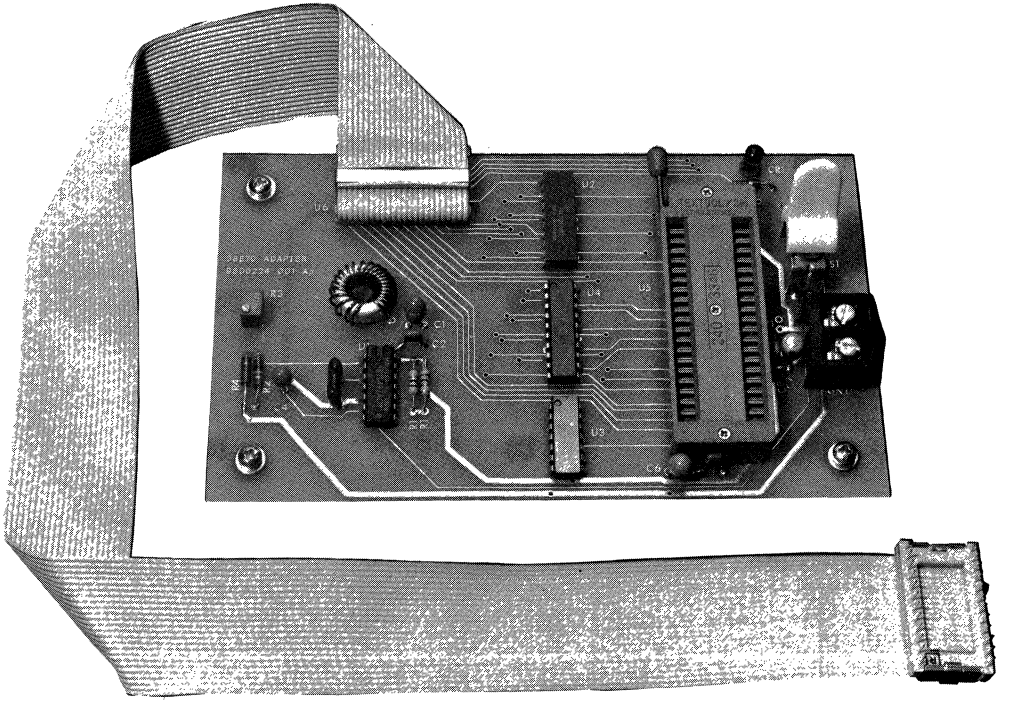
DESCRIPTION

The Fairchild F38E70 Programming Board enables the F38E70 single-chip microcomputer to be programmed without special software or hardware on a PROM programmer that can program 2716-type EPROMs. When inserted into the programming board, the F38E70 is recognized by the programming unit as a 2716-type EPROM.

The programming board is linked to the program unit by means of a cable having a 24-pin connector that plugs into the 2716 programming socket.

- **Enables Programming of F38E70 Single-Chip Microcomputers on PROM Programmer Units Having a 2716-Type EPROM Capability.**
- **No Special Software or Hardware Required.**
- **On-Board DC-to-DC Converter for V_{PP} Generation.**
- **LED Indicates V_{PP} On.**
- **External 5V Power Supply Connector Provides Continuous V_{CC} for Cold Programming Sockets.**

F38E70 Programming Board



9

F38E70
Programming Board

DEVICE OPERATION

The program or verify mode is entered when V_{PP} is applied to TEST 1/V_{PP} (pin 21) of the F38E70 microcomputer. The address of the location to be programmed is applied to ports 0 and 1, and the inverse of the data to be programmed is applied to port 5. The PROG pin (pin 24) is then pulled low for 50 ms and the data written into the designated location. (Operation of the board cannot be guaranteed when used with PROM programmers that do not utilize this method.)

The programmer board uses a 74LS240 to invert the data from the EPROM programming unit. A 74LS244 is used to buffer the data output of the F38E70 from the data lines of the programming unit. Figure 1 shows the pin connections for the programming board, the buffer and inverter, and the device socket.

An on-board dc-to-dc converter generates V_{PP} voltage, which must be present during the F38E70 program verification mode. This voltage is adjusted to the required value by varying an on-board potentiometer (R3). A switch (S1) is provided to disconnect the V_{PP} voltage when inserting or removing the F38E70, and an LED (CR1) is used to indicate the presence of the V_{PP} voltage on the F38E70. The switch must be off and the LED indicator must not be lit when inserting or removing the F38E70.

The programming board logic requires constant V_{CC} voltage; therefore, an external power supply must be used with EPROM programming units that remove the V_{CC} voltage from the EPROM socket when not in the program or verify mode. The external power supply is connected to TB1 on the programming board after removing the V_{CC} jumper (W1).

Caution

Applying V_{PP} voltage to the TEST 1/V_{PP} pin without the presence of V_{CC} will damage the F38E70.

PROGRAMMING VOLTAGE ADJUSTMENT

To adjust the V_{PP} voltage to the required level (specified in the information sheet accompanying each F38E70 unit), apply + 5 V to the programming board and set S1 to turn on the V_{PP} voltage. Measure the voltage between pin 21 of the programming socket and ground. Adjust the potentiometer to give the correct V_{PP} voltage.

PROGRAMMING PROCEDURE

The following procedure is used to program the F38E70.

1. Erase the F38E70 under a suitable UV light source.
2. Configure the PROM programmer for 2716-type EPROMs, and load the data to be programmed.
3. Insert the programming board cable connector into the appropriate socket of the programmer; ensure that pin 1 of the connector mates with pin 1 of the socket.
4. Turn on the external + 5 V power supply, if used.
5. Set S1 to extinguish the LED (CR1).
6. Insert the F38E70 into the programming board socket, ensuring that pin 1 of the F38E70 is located near the handle of the zero-insertion-force socket.
7. Set S1 to turn on CR1.
8. Perform the 2716-type EPROM programming operation on the PROM programmer.
9. When programming is complete, switch off S1, verify that CR1 is extinguished, and remove the F38E70 from the programming board socket.
10. Cover the window on the F38E70 with suitable material to prevent accidental erasure.

DC Characteristics

V_{CC}	+ 5 V ± 5%
Current Consumption	< 400 mA

Ordering Information

Order Code	Description
9800224-001	F38E70 Programming Board

FAIRCHILD
A Schlumberger Company

F8 and F387X Formulator

Microprocessor Product

Description

The microprocessor system designer can create hardware and software development systems for the F8 and F387X by selecting modular subassemblies from Fairchild's line of F8 and F387X design aids. Development may start with a Formulator Mark I single-board system, then expand to more sophisticated Mark II or Mark IIFD development systems than can handle both software and hardware development (see figure 10-1). Future growth may lead to a complete Formulator Mark III with intelligent control panel, power supply and accessories, or to the top of the line Formulator Mark IIIFD with floppy disk drives.

Three growth packages plus a selection of optional modules provide a practical method for upgrading the single-board Mark I to either the Mark II or Mark IIFD, or to the maximum system configuration Mark III or Mark IIIFD. Using the growth packages, the designer can begin sophisticated system application programs at very low cost and then upgrade the development tools in relatively inexpensive steps at a later time.

The most elementary configuration, called the Formulator Mark I, includes a processor module that contains an F8 CPU, program storage unit that includes a debug program, dynamic and static memory interface circuits, 1024 bytes of random access memory, and the necessary buffers and other components for hardware development. It also includes a 13-slot card cage, an I/O cable kit, and a power cable.

The second level, the Formulator Mark II, includes all of the Mark I components plus a memory board with 16 kilobytes of RAM and the complete Formulator operating system, designated FOS. The FOS provides complete software development capability, including an assembler, editor, and debug package, and drivers for a teletype or the TI Silent 733 terminal.

The third level, the Formulator Mark IIFD, is identical to the Mark II with the addition of interface cards and cables for an iCOM FD3712 dual-drive floppy disk system and Fairchild DOS4 Floppy Disk Operating System.

The fourth level, the Formulator Mark III, includes an intelligent control panel, a serial communications module, a quad I/O module, an attractive cabinet, and a power supply. Also included are 16K bytes of RAM, the Formulator processor module, and the Formulator operating system.

The top of the line is the Formulator Mark IIIFD. This system is identical to the Mark III, except it interfaces to the iCOM dual-drive floppy disk.

Three growth packages are available for Mark I, Mark II, and Mark III expansion. Growth Package I upgrades the Mark I system to the Mark II level. Growth Package II converts the Mark II to the full Mark III level. Growth Package III upgrades either the Mark II or Mark III to the Mark IIFD or the Mark IIIFD floppy disk configurations.

Other boards are available as options for all five Formulator configurations to increase the flexibility of the units by adding to their capabilities. These include 4K-byte RAM, 4K-byte PROM, and 16K-byte RAM boards, as well as an I/O light board, a communications board with UART, a byte-parallel board for peripheral interface, and a PROM programmer.

FAIRCHILD
A Schlumberger Company

FS-I
Fairchild System-I

Advance Product Information | Microprocessor Product

Description

The Fairchild System-I (FS-I) is a versatile, multi-user development system designed to support software development and hardware prototyping for applications using Fairchild microprocessors, including the F8, F3870, F6800, F6809, F9445, F16000, and such upcoming microprocessors as the F9450.

Three principal versions of the FS-I are available: The FS-I Standard System, the FS-I Multi-User System, and the FS-I Entry-Level System. Numerous software and hardware options are available that operate under Fairchild's Interactive Multi-User Disk Operating System (IMDOS). The FS-I also supports the in-circuit emulation and tracing (EMUTRAC™) system for the F3870, the F6800, F6809, and the F9445 microprocessors. (For a description of the EMUTRAC system, see *EMUTRAC Advance Product Information.*)

Standard System

System features include:

- **CPU with 128K-Byte RAM and F9445 Instruction Set.**
- **A Winchester and a Double-Density Floppy Drive Provide Approximately 10M-Byte of Mass Storage.**
- **I/O Controller Board Provides Winchester/Floppy Disk Controller Interface.**
- **Nine Asynchronous Serial RS-232C Ports (Up to 19.2K Baud) Provide Support for CRT Terminal, Optional Letter-Quality Printer, Modem, and Other Serial Devices.**
- **One Synchronous Serial RS-232C Port (Up to 19.2K Baud) and Selectable Protocols, such as BISYNC, DDCMP, SDLC, and HDLC.**
- **PROM Programmer Port to Interface to the Optional Fairchild PROM Programmer Unit.**
- **Parallel Printer Port (Centronics-Compatible Interface).**
- **Programmable Real-Time Clock.**
- **One CRT Terminal.**
- **Single-User Version of IMDOS, System Processors, and System Utility Programs (see "System Software").**
- **BASIC Language Interpreter with Interface to Custom F9445 Assembly Language Programs.**
- **FS-I Diagnostic Programs.**
- **Provides Full Support for the F9445 and for the PEP 45 Microcomputer System.**
- **Hardware and Software Upgradable to Multi-User System.**
- **EMUTRAC Can Be Added to the Standard System.**

Multi-User System

System features include:

- **Fully Equipped for Four Timesharing Users (Expandable to Eight Simultaneous Users with Additional Terminals and Cables).**
- **A 16-Bit CPU with 128K-Byte RAM and F9445 Instruction Set.**
- **A Winchester and a Double-Density Floppy Drive Provide Approximately 10M-Byte of Mass Storage.**
- **Memory Management and Protection Unit (MMPU) Board with 384K Bytes of RAM (Gives the System 512K Words of RAM).**
- **I/O Controller Board Provides Winchester/Floppy Disk Controller Interface.**
- **Nine Asynchronous Serial RS-232C Ports (Up to 19.2K Baud) Provide Support for CRT Terminals, Optional Letter-Quality Printer, Modem, and Other Serial Devices.**
- **One Synchronous Serial RS-232C Port (Up to 19.2K Baud) and Selectable Protocols, such as BISYNC, DDCMP, SDLC, and HDLC.**
- **PROM Programmer Port to Interface to the Optional Fairchild PROM Programmer Unit.**
- **Parallel Printer Port (Centronics-Compatible Interface).**
- **Programmable Real-Time Clock.**
- **Four CRT Terminals.**
- **Multi-User Version of IMDOS, System Processors, and System Utility Programs (see "System Software").**
- **BASIC Language Interpreter with Interface to Custom F9445 Assembly Language Programs.**
- **FS-I Diagnostic Programs.**
- **Provides Full Support for the F9445 and for the PEP 45 Microcomputer System.**
- **EMUTRAC Can Easily Be Added to the Multi-User System.**

Entry-Level System

System features include:

- **A 16-Bit CPU with 128K-Byte RAM and F9445 Instruction Set.**
- **Two Double-Density Floppy Disk Drives Provide Approximately 1M-Byte of Mass Storage.**
- **I/O Controller Board Provides Floppy Disk Controller Interface.**
- **Nine Asynchronous Serial RS-232C Ports (Up to 19.2K Baud) Provide Support for CRT Terminal, Optional Letter-Quality Printer, Modem, and Other Serial Devices.**

9

™ EMUTRAC is a trademark of Fairchild Camera and Instrument Corp.

- **One Synchronous Serial RS-232C Port (Up to 19.2K Baud) and Selectable Protocols, such as BISYNC, DDCMP, SDLC, and HDLC.**
- **PROM Programmer Port to Interface to the Optional Fairchild PROM Programmer Unit.**
- **Parallel Printer Port (Centronics-Compatible Interface).**
- **Programmable Real-Time Clock.**
- **One CRT Terminal.**
- **Single-User Version of IMDOS, System Processors, and System Utility Programs (see "System Software").**
- **BASIC Language Interpreter with Interface to Custom F9445 Assembly Language Programs.**
- **FS-I Diagnostic Programs.**
- **Full Support for the F9445 and for the PEP 45 Microcomputer System.**
- **Hardware and Software Factory-Upgradeable to Standard or Multi-User System.**
- **EMUTRAC and MMPU Can Be Added to System.**

System Hardware

The hardware comprising the FS-I development system is housed in a single enclosure that contains the mainframe CPU, I/O board, optional boards, and disk drives.

The mainframe consists of:

- **Single-Board 16-Bit CPU with 128K Bytes of RAM, 4K-Byte PEPBUG45 PROMs for Bootstrapping the System, Real-Time Clock, an RS-232C-Compatible Port, and a Centronics-Parallel Compatible Port.**
- **Power Supplies.**
- **I/O Controller Board with the Following:**
  - **Eight Asynchronous Serial RS-232C Ports, with Four Ports Having Full Modem Control and All Ports Having Data Rate Selectable Up to 19.2K Baud, that Allow Timesharing by Up to Eight Concurrent Users on Systems Equipped with MMPU Board and Multi-User Operating System Software.**
  - **One Synchronous Serial RS-232C Port (Up to 19.2K Baud) and Selectable Protocols, such as BISYNC, DDCMP, SDLC, and HDLC.**
  - **A Parallel Data Channel Interface Compatible with Shugart Associates System Interface for Communicating with Disk Units.**
  - **8-Bit Parallel Port to Interface with Optional Fairchild PROM Programmer.**
- **A Total of Nine Asynchronous Serial Ports (RS-232C-Compatible, DB25-Pin Female Connectors).**
- **One Parallel Printer Port (Centronics-Compatible Interface, DB25-Pin Connector).**
- **Expansion Slots for Fairchild's Optional I/O Controller Boards, Optional EMUTRAC Controller Board, Memory Expansion Boards, MMPU Board, and Industry-Standard, Nova® I/O-Compatible Interface Boards.**
- **Depending Upon System Configuration, the Mainframe Contains a Single 10M-Byte Winchester and a Single 0.5M-Byte Double-Density Floppy Disk Drive or Two 0.5M-Byte Double-Density Floppy Disk Drives.**

The MMPU board expands the physical address space of the FS-I to 4M words by performing logical-to-physical address translation. This board is required for multi-user system software. With its 384K bytes of RAM, the MMPU board extends the FS-I memory to 256K words.

Hardware Options

The FS-I systems support the following Fairchild-supplied hardware options:

- **Additional I/O Controller Boards that Provide Asynchronous RS-232C Ports (Up to 19.2K Baud) in Sets of Eight, a Synchronous RS-232C Port for each I/O Controller Board, Data Channel Interface to Disk Units, and a PROM Programmer Port for each I/O Controller Board.**
- **Fairchild's PROM Programmer Unit.**
- **MMPU Board that Provides Memory Mapping and Protection Expansion in Increments of 384K Bytes, Optional Multi-User Software Allows the MMPU Board to Support Eight Simultaneous Users.**
- **Memory Expansion Board that Provides 384K Bytes of Additional RAM (Requires an MMPU Board in the Chassis).**
- **EMUTRAC System Controller Board that Provides the Hardware Interface Between the CPU Board in the FS-I and Processor-Specific EMUTRAC Modules.**
- **EMUTRAC Modules and EMUTRAC Control Software that Support the F3870, the F6800, the F6809, and the F9445 Microprocessors.**
- **Additional CRT Terminals.**
- **Dot Matrix Printer—Texas Instruments Model 810 Basic RO Terminal (150 CPS), Centronics Parallel Interface, and Cable.**
- **Daisywheel Letter-Quality Printer—Qume Model Sprint 9145 with Bidirectional Forms Tractor (45 CPS), Serial Interface, and Cable.**

® Nova is a registered trademark of Data General Corp.

System Software

The interactive multi-user disk operating system (IMDOS) is the principal operating system for the FS-I. In addition to being an operating system, the IMDOS includes the following features that are useful for developing F9445-based systems:

IMDOS	Single-User Supervisor—The supervisor manages the FS-I resources and controls the I/O. Multi-User Supervisor—The supervisor manages the FS-I resources for up to eight simultaneous users, controls the I/O, and interfaces transparently to the MMPU board (included only with the multi-user system). Executive—The executive provides the command language interface between the user and the supervisor.
EDIT	The EDIT program provides the ability to create and modify text files.
MACRO	The MACRO program is the macroassembler for F9445 macro assembly language.
RELOAD	The RELOAD program is used to link relocatable macro assembly language programs to create executable F9445 absolute assembly language programs.
PEPBUG-45	The PEPBUG-45 program is a virtual console and debugging tool for F9445 absolute assembly language programs. The PEPBUG-45 program is also available in PROM.
PEPLINK	Provides capability to download programs from the FS-I to PROM or RAM on the PEP 38, PEP 45, and PEP 68 microcomputer systems.
Utility Library	Implements the utility functions listed in the FS-I User Guide.
PHONE	The PHONE program establishes communication between the FS-I and a modem or telephone line. Software switches govern communication protocols.
SCRIPT	The SCRIPT program processes a text file that contains SCRIPT commands to produce an aesthetically pleasing document.
TYPESET	The TYPESET program processes a text file that contains TYPESET commands to produce an aesthetically pleasing document.
TESTS	A series of programs that test the FS-I hardware. The diagnostic programs are available on diskette in a version suitable for downloading to an F9445-based system.
BASIC	Language interpreter with interface to custom F9445 assembly language programs.

This powerful software package, which is included with the standard, multi-user, and entry-level systems, offers advanced capabilities that the user would normally expect from a much larger system, such as:

- **Multi-User Timesharing**
- **System Executive, Including File Management System with Version Numbers for Automatic Backup**
- **Memory Management and Protection by Memory Mapping**
- **Password Protection**
- **Interactive Command Language and Command Files**
- **Multiple Directory Devices**
- **Device-Independent I/O**
- **Hard Disk, Magnetic Tape, Modem, and Real-Time Clock Support**
- **Documentation Aids**
- **Concurrent Processing and Spooling**

Software Options

F9445 MICROFORTRAN	An extended subset of FORTRAN66 that interfaces with custom F9445 assembly language subroutines. MICROFORTRAN produces "ROMable" F9445 code and can be operated under the real-time executive (REX).
F9445 PASCAL	A Jensen and Wirth-compatible PASCAL. The F9445 PASCAL compiler generates F9445 code and interfaces with custom F9445 assembly language subroutines.
FS-I/PEP 38 System Software	Includes F8/F3870 cross assembler and program for downloading to the PEP 38 system.
FS-I/PEP 68 System Software	Includes F6800 cross assembler, F6809 cross assembler, F6800-to-F6809 translator program, and program for downloading to the PEP 68 system.
F16000 Cross Software	Assembler, debugger, and downloader allow the FS-I to generate 16000 code that can be downloaded to an F16000-based system.
F9445 REX	A real-time executive for F9445-based systems. The REX system allows creation of custom REX programs, linkable using RELOAD.
F9445 PEPBASIC	A diskette version of PEPBASIC (supplied on PROM with the PEP 45 system). A 2K-word subset of BASIC, which accepts abbreviations, that is extendable with custom F9445 assembly language subroutines.
EMUTRAC Control Software	Optional EMUTRAC control software packages provide support for each processor-specific EMUTRAC module. (Refer to *EMUTRAC Advance Product Information.*)

In addition, all Fairchild software for the FS-I is independently available without system purchase under an appropriate software license agreement.

FS-I Command and Utility Summary

ABTOSV fs1, fs2	Converts an absolute binary file into a save file.
APpend fs1, fs2	Appends a copy of fs1 to the end of fs2.
APR/* ...	Special print routine
ARITH45	Tests arithmetic instructions.
BACKUP	Backs up a file or a group of files.
BASIC	Invokes the F9445 BASIC interpreter.
BINCOM/* (fs1, fs2)	Compares two or more files.
CH/* fs	Makes an entry in the change log before editing a file.
CLEANSE/* (fs)	Removes all but the highest version of all files.
CLear	Closes any open files and clears I/O channels.
COMPILE	Invokes the F9445 Pascal compiler.
CONFIGURE	Creates a custom version of IMDOS.
Copy fs1, fs2	Copies file1 to file2.
COPYMINSYS	Copies a minimum system from unit 0 to unit 1.
CP (fs1, fs2)	Copies files and compares original and copy.
DElete fs*(/*)	Deletes files. Allows wildcards.
DIrectory (n:)(fs*)	Lists file(s) and chain table. Allows wildcards.
DUmp fs(/*)	Dump a file in octal or decimal.
EDIT	Invokes the editor.
EDit fs	Edits a particular file.
EMUTRAC T:	Invokes EMULTRAC
EXER45	Exercises the FS-I hardware.

F38ASM	Invokes the F3870 cross assembler.
F68ASM	Invokes the F6802/F6808 cross assembler.
F69ASM	Invokes the F6809 cross assembler.
FIND/* (<in) (>out)	Gets information from a keyed source file.
FORMAT	Prepares a disk to become a system or data disk.
FORTRAN	Invokes the MICROFORTRAN compiler.
GET n	Displays contents of location n (octal).
GOodbye	Loss user out.
GREP/* pattern...	Finds and outputs lines that match the pattern.
HELP/* (topic)...	Shows help information.
INput fs	Reads and executes commands from a file.
LIBEDIT/* ...	Manipulates libraries of relocatable binary files.
LIst (n:) (fs*)	Lists files on a device. Allows wildcards.
LISTPASS-WORDS	Lists user names, passwords, default security (2-only).
LOad fs	Loads a file into memory.
LOGin user-	Loss in a user; password required.
MAC/* (<in) (>out)	General macro processor
MACRO/* ...	Invokes the F9445 macroassembler.
MEM45	Tests the FS-I memory.
MINIREPAIR	Checks and repairs directory structure on a disk.
MISC45	Tests miscellaneous F9445 hardware instructions.
MLIMIT	Tells the IMDOS the highest location (in lower 32K words of memory) used by a program.
MPMACS IN	Includes file for creating manual pages.
OEDIT fs	Examines and allows modification of disk files.
ONline m,n,x,...	Places devices on line and specifies order of search.
OUtput fs	Directs output of LIst, DIrectory, SPace commands to a file.
PASSWORDS	Creates user names, sets passwords, etc. (2-only).
PAUSE	Halts a program and waits for a (RETURN) to continue.
PEEK	Examines status of the IMDOS, including use of buffers.
PEPBUG45	Invokes the PEPBUG45 debugger.
PHONE (/D) port	Enables communication with a device attached to a port.
PRE/* ...	Preprocesses file(s).
PUT n	Enters octal number into memory location opened by GET.
PRint fs	Prints a file on the line printer (L:).
QCOPY/* (fs)	Prints a file on the letter-quality printer.
READ.ME	Type this file for information about a release.
RELOAD/*	Invokes the linker/loader.
REname fs,new—fs	Renames a file.
RESTORE	Returns programs created with BACKUP.
RUN fs	Loads and executes a file.
fs	Executes fs.SV; if no .SV, executes fs.CM.

SAVE fs,loc:....,addr	Saves the contents of memory in a file.
SCRIPT in,out,los	Creates a formatted document.
SEcurity fs/code	Changes the security protection on a file.
SORT/* ...	Sorts a file (about 500-line limitation).
SPace	Shows space useage on disk.
STart addr	Starts a program in memory at an address.
SVTOAB	Converts a save file into an absolute binary file.
TApe n(.)	Sets record length on a tape; octal or n. = decimal.
TAPEREAD	Reads various tape formats.
TAPEWRITE	Writes tapes in various formats (EBCDIC etc.).
TErminal type	Identifies terminal type to the IMDOS.
TIMER45	Runs instruction set timer.
TRANS09	Translates an F6802/F6808 source to F6809.
TYpe fs	Writes contents of a file on the screen.
TYPESET	Creates a formatted document.
WRITESYS	Creates/updates the IMDOS on a system disk.

Dimensions and Power Requirements

The FS-I standard mainframe enclosure measures only 26 inches long by 19 inches wide by 13 inches high. It requires a 115 V, 60 Hz ac power source. A 50 Hz system is also available.

FAIRCHILD
A Schlumberger Company

PEP 38
Prototyping, Evaluation and Programming Board

Advance Product Information

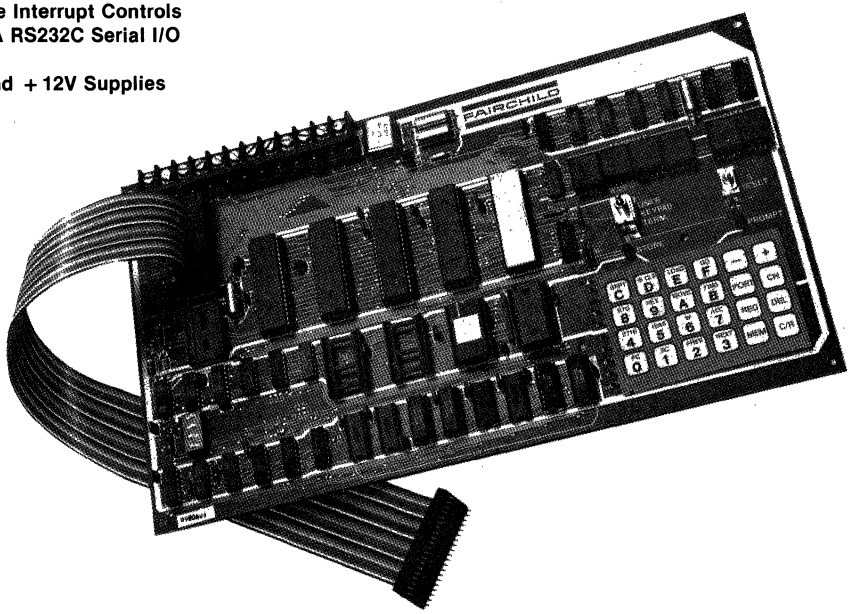
Microprocessor Product

Description

A single-board microcomputer for program development timing, debugging, and emulating the F387X family of single-chip microcomputers, the PEP 38 system includes and F38E70 EPROM microcomputer programmer, an on-board keypad, address and data displays. A 40-pin emultation cable is also provided.

Features

- **Full In-Circuit Emulation of the F3870 and F3872 Microcomputers**
- **On-Card Keypad for Command and Data Entry**
- **On-Card 7-Segment Address and Data Displays**
- **Programming Sockets for F38E70s and 2716s**
- **2K Bytes of 2114 Static RAM Plus Space for an Additional 2K Bytes**
- **Space for 6K Bytes of 2716 EPROM**
- **2K-Byte Firmware Monitor**
- **Flexible Memory-Map Strapping Options**
- **Crystal-Controlled System Clocks**
- **Four General-Purpose Programmable Timers**
- **Four General-Purpose Interrupt Controls**
- **Current-Loop and EIA RS232C Serial I/O**
- **Spare 8-Bit I/O Port**
- **Requires Only +5 and +12V Supplies**



9

FAIRCHILD

A Schlumberger Company

PEP-45
Prototyping, Evaluation and Programming Board

Advance Product Information

Microprocessor Product

Description

The Fairchild PEP-45 is a single-board microcomputer for Prototyping, Evaluation, and Programming of microprocessor-based system applications using the F9445 microprocessor. When used with the Fairchild System-I (FS-I) development system, the PEP-45 board provides capability for executing and debugging software directly on the F9445 microprocessor.

- **Stand-alone Prototyping, Evaluation, and Programming Board.**
- **Provides a Powerful Development Tool to Support F9445 Microprocessor-based System Development.**
- **Utilizes All the Advantages of the F9445 Microprocessor, with its Powerful Instruction Set and High Throughput.**
- **Memory Options for Bipolar and NMOS Memories.**
- **Interfaces with IEEE 796 Standard Bus.**
- **Buffered F9445 bus.**
- **On-board EPROM Programmer.**
- **Adapts to 16K or 32K Byte EPROMs or 64K Byte Masked ROMs.**
- **Standard- and High-Speed RAM Options.**
- **Console Commands.**
- **Two Serial I/O Ports.**
- **16-Bit Parallel Input/Output.**
- **Four Interrupt Sources.**
- **Five Status Lines.**
- **On-board +12 V and +25 V Voltage Converter.**
- **Requires Single +5 V Power Supply.**

The PEP-45 board is primarily intended for use in hardware prototyping and software development applications. It may also be tied to a host computer, such as the FS-I, for large program editing, assembling/compiling, and general file storage and handling. Cross-assembler software packages are available for creating machine-executable programs in formatted form. These programs may be down-loaded from the host computer system into the PEP-45 board via one of the two serial I/O channels. Since the PEP-45 board can operate in a transparent fashion, it may be placed in-line between the user's in-house terminal and the host computer, giving the PEP-45 the power of the host.

Also useful for incoming inspector of F9445 parts and as a microcomputer training tool, the PEP-45 interacts with the user at the control terminal, with prompts that assist programming. The control terminal may be a video terminal, printer terminal, or from a microcomputer control console.

Software Support

In addition to serving as an efficient stand-alone evaluation module, the PEP-45 is designed to operate as a key module of the FS-I development system. A PEPLINK utility transparently couples the FS-I video terminal to the PEP-45 board.

A powerful PROM-based PEPBUG debugging monitor provides commands for trouble-shooting assembly language programs and for developing and testing peripheral circuits and custom interfaces. A PROM-based PEPBASIC language allows programming in a high-level language.

Hardware Specifications

Microprocessor	
CPU	F9445
Data word size	16 bits
Instruction word size	16 bits
Address capability	128K bytes
Console controller	F9470
Memory	
RAM	8K bytes (4K words) static RAM (or optional high-speed RAM)
ROM	Eight sockets for 16K bytes of F2716 EPROMs (8K words), or up to 32K bytes using F2732 EPROMs (16K words), or masked 64K byte ROMs using F3564
Expansion	External memory in any combination of RAM or ROM up to 64K bytes maximum (in 16-bit-wide only)
Input/Output	
Parallel I/O	Two TTL-compatible, 16-bit I/O ports (one input, one output)
Serial I/O	Two programmable, asynchronous channels, with RS-232 interfaces. Each channel is software-selectable to a baud rate of 110, 300, 1200, 1800, 2400, or 4800 baud
Real-Time Clock	Continuously selectable real-time clock interrupts from approximately 200 μs to 200 ms

9

System Buses

Dual backplane buses	PI — An 86-pin asynchronous system bus compatible with standard Multibus 16-bit slave boards and multi-master option
	P2 — A 60-pin buffered F9445 bus that allows complete expansion of processor capabilities and faster operating speeds
I/O buses	J1 — A 9-pin RS-232C serial I/O interface for control terminal
	J2 — A 9-pin RS-232C second serial I/O interface for a serial printer or a host computer
	P3 — A 40-pin applications connector with two parallel I/O ports (one input and one output), and with status and control bits. May be used for connection to the microcomputer control console or to a high-speed parallel printer (Centronics-type)
Connectors	P1 — An 86-contact, double-sided edge connector on 0.156″ centers
	P2 — A 60-contact, double-sided edge connector on 0.100″ centers
	J1, J2 — 9-pin, D-type subminiature right-angle connectors
	P3 — A 40-pin, double-sided edge connector on 0.100" centers.

Power Supply Requirements[1]	+5 V ±5% at 3.5 A (typ)
Environmental Requirements	
Temperature	0°C to +50°C
Humidity	0% to 90% (noncondensing)
Physical Envelope Dimensions[2]	
Height	10.0 (254)
Length	12.0 (305)
Thickness	0.75 (19.05)
Weight	17 oz. (approximately)

Notes
1. Power may be applied to the board either through the card-edge backplane connector or by connection of discrete wires to the board.
2. All dimensions are in inches and millimeters (in parentheses).

Ordering Data

Part Number	Product Code	Description
PEP 9445SFX	A F944516PEP	PEP-45 Board with 8K byte PROM sockets populated with PEPBASIC and PEPBUG firmware. Firmware carries copywriter notice. Minimum of four PROM sockets will not be populated. PEP-45 Users Guide, PEPBASIC, and PEPBUG Users Guide supplied.
PEP 9445SXX	A F944516PEP	PEP-45 Board with 8K byte static MOS and eight PROM sockets not populated. PEP-45 Users Guide supplied. No firmware included.
PEP 9445HXX	A F944520PEP	PEP 9445 Board with 8K byte high speed RAM and PROM sockets not populated. PEP-45 Users Manual supplied. No firmware or users guides included.

FAIRCHILD
A Schlumberger Company

PEP 68 System Single-Board Microcomputer Development System

Advance Product Information

Microprocessor Product

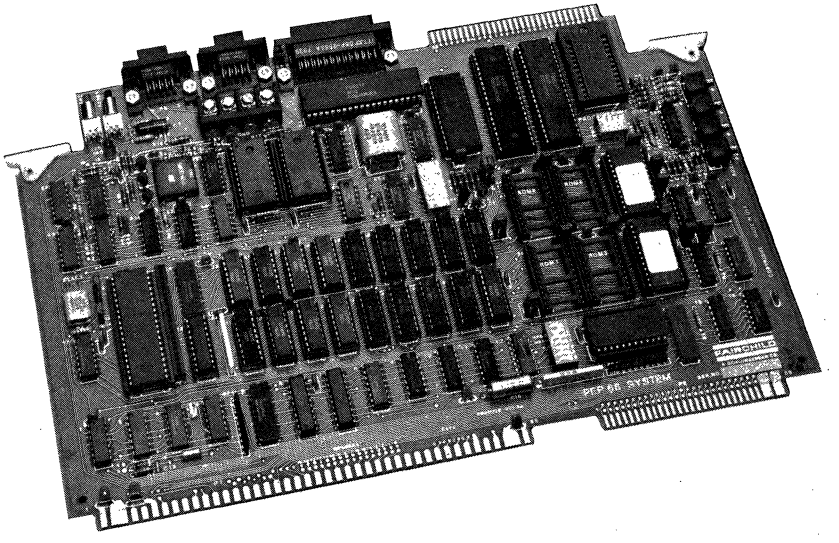
Description

The PEP 68 System is a single-board microcomputer specifically designed to aid microprocessor hardware/software designers in designing, prototyping, and debugging their 6802-, 6808-, or 6809-based system. A powerful, ROM-based debugging monitor provides commands for trouble-shooting machine-language programs. Other monitor commands provide for easy development and testing of peripheral circuits and custom interfaces. The monitor includes a full complement of utility routines to make the hardware/software/firmware design cycles as easy as possible.

The PEP 68 System is useful as a microcomputer training tool. Its friendly interaction with the user at the control terminal, through its liberal use of prompting, makes procedures easy to learn for the beginner. The system can be operated using only a serial display terminal and a power supply.

Since the system possesses two separate bus connectors, expansion with external memory or peripheral boards is simply a matter of providing a backplane connection. Thus, the PEP 68 System can act as a *bus master* in a multicard system.

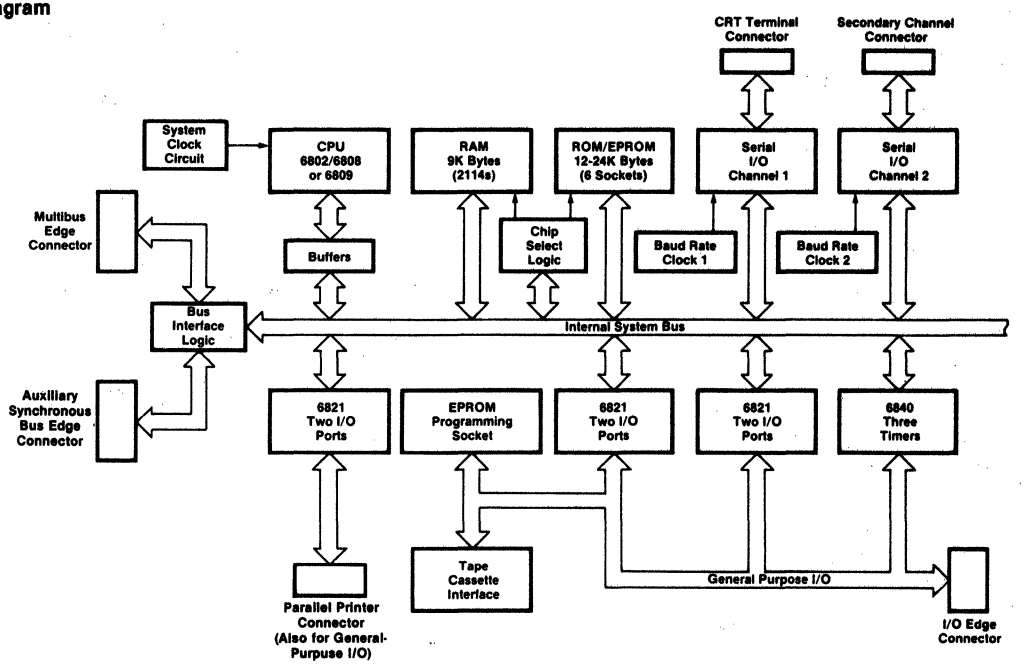
- **Single-Board, Stand-Alone System**
- **Processor Options—6802, 6808, or 6809**
- **Asynchronous Multibus* Compatible**
- **Auxiliary Synchronous 680X Bus**
- **Programming Socket for 2716 or 2732 EPROMs**
- **8K-Byte System Monitor in ROM**
- **9K Bytes of Static RAM—8K User, 1K System (Write-Protectable Segments)**
- **Six Sockets for User-Supplied ROM/EPROM (2K, 4K, or 8K Types)**
- **Sixteen Possible Memory Map Configurations (Switch-Selectable)**
- **Two High-Speed Audio Cassette Tape Interfaces**
- **Two Independent Serial I/O Channels—RS-232-C**
- **Independent Baud Rate Selection—50 Through 19.2K bps**
- **Connector for Parallel Printer (Centronics Type)**
- **Six 8-Bit Parallel I/O Ports Plus Controls**
- **Three Programmable 16-Bit Binary Timers**
- **+5 Volt-Only Operation**



9

*Multibus is a trademark of Intel Corporation.

Block Diagram



Hardware Description

The PEP 68 System is a single-board, stand-alone microcomputer utilizing either a 6802, 6808, or 6809 microprocessor as its central processing element. The system may be connected to a larger, host computer system to utilize that system's file storage, editing, and assembling capabilities. Thus, source language programs can be created, edited, and assembled on the host computer system using PEP-UP cross-assembler software packages. The resulting machine-language programs can then be downloaded into the PEP 68 System on-board RAM, executed, and debugged.

Dual Bus Interfaces

The PEP 68 System can also serve as the central processor in a multiboard system with connections to peripheral boards accomplished via a bus interface and the cardcage backplane. It has two separate bus interfaces: one synchronous and one asynchronous. The synchronous bus interface includes the system CPU signals and allows for expansion using synchronous 680X peripheral boards. The asynchronous bus interface, on the other hand, allows for system expansion with peripheral boards that use the industry-standard Multibus interface. The asynchronous aspect is accomplished by stretching the CPU's system clock. Both bus interfaces on the PEP 68 System can simultaneously connect to external peripheral boards. However, the PEP 68 System can be the only *master* central processor board in the system.

Memory

The PEP 68 System contains 8K bytes of on-card static RAM storage for user application programs. Each 4K-byte segment can be separately write-protected by means of either an on-card switch or by signals at the bus edge connector. There are an additional 1K bytes of RAM for use by the board's ROM monitor. A total of six ROM or EPROM sockets is provided on-board. Each can be jumpered for either 2K, 4K, or 8K-byte devices, i.e., many of the various 24-pin ROMs or EPROMs. Normally, one or two of these sockets contain the FAIRBUG/68 monitor ROM, but if desired, they can be used for user code instead.

All on-board memory and I/O address decoding is done through the use of a bipolar PROM. This PROM and the four DIP switches tied to it allow the user to select one of 16 different memory map configurations depending on system requirements. This feature is especially useful during the program development phase of a project since the user's code can be resident in either RAM or EPROM and can be relocated with a switch change.

Serial I/O Channels

The board contains two serial I/O channels. Both channels are general-purpose and may be used with any serial RS-232-C device. One channel is normally used for communication with the user's control terminal; while the second channel would normally be used for the interface to the host computer system. However, this second channel could be used for any serial I/O use, such as a printer or modem. The RS-232-C interfaces generate their own +12 and −12 volt levels. Thus no additional supplies, other than the +5 V_{dc} supply, are required.

Each channel has a separate baud-rate clock circuit in which the baud-rate is hardware switch-selectable. This allows very fast communication with a local command terminal on channel 1 and communication with a slower speed device such as printer, modem, or phone link on channel 2. Allowed baud rates on each channel are 50, 110, 150, 300, 1200, 1800, 2400, 4800, 9600, or 19,200 bps.

Parallel I/O Ports and Programmable Timers

The PEP 68 System has six I/O ports and associated control signals that can be used for general-purpose input/output. Four ports are available at the top card edge connector, while the remaining 2 are accessible via a special plug-in connector. The latter two ports can be used optionally for driving a high speed parallel printer with a special cable that attaches to the board through the plug-in connector.

The signals and controls associated with the three binary timers are accessed via the card edge connector. Each of these three software programmable timers is 16 bits long and can be operated in several different modes, including continuous, single-shot, frequency comparison or pulse-width comparison modes.

Audio Cassette Tape Interfaces

There are provisions on the PEP 68 System for connecting two audio cassette tape recorders for storing and retrieving user's applications programs. The interfaces are driven by code contained within the monitor, thus minimizing the required hardware circuitry. The recording format is a self-clocking method that allows synchronous data transfers rates of up to 2000 bits per second. Connections between the recorder and the board are made with subminiature phone jacks.

EPROM Programming Socket

The PEP 68 System provides a zero-insertion-force socket for electrically programming 2716 and 2732 type EPROMs; therefore, the user's application programs residing in RAM can be preserved by "burning" the code into an EPROM. Subsequent execution can be from either RAM or EPROM. The programming socket is driven by signals from three of the six on-board I/O ports. The monitor provides commands to perform the following: blank check tests, copy EPROM contents to memory, verify EPROM contents against memory, program any portion of EPROM, and masking non-blank EPROMs against code in memory.

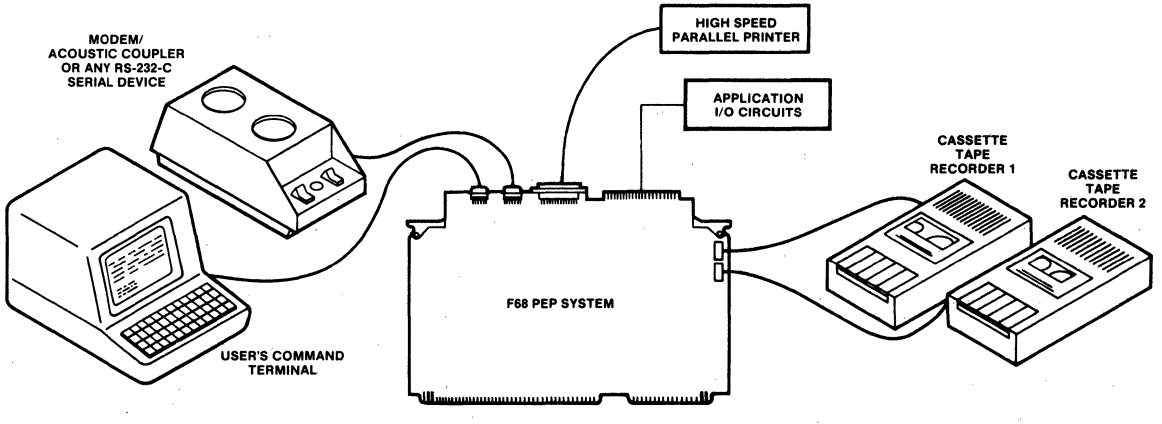
Software Description

- **Display or Alter any CPU Register**
- **Display or Alter any Memory Location**
- **Display a Range of Memory**
- **Display the Previous (or next) Location in Memory**
- **Rapidly Input Consecutive Data Strings to Memory**
- **Find (search for) the Address of the Next Occurrence of a Specified Data String**
- **Fill a Range of Memory with a Given Data String**
- **Move (Copy) a Block of Memory from One Address Range to Another Address Range**
- **Go to an Address and Begin Executing a User Program**
- **Load a Formatted File from Either Serial Channel with an Optional Address Bias (Multiple Formats Allowed)**
- **Punch/Dump a Formatted File to Either Serial Channel with an Optional Address Bias (Multiple Formats)**
- **Compare Two Memory Ranges for Differences**
- **Calculate Checksums Over a Range of Memory**
- **Insert a Program "Patch"**
- **Disassemble Machine Code into Assembly mnemonics**
- **Set, Clear, and Display up to 8 Address Breakpoints**
- **Remove all Breakpoints Temporarily and Then Be Able to Restore Them Intact**
- **Continue or Resume Execution After a Break Occurs or After Stopping or Tracing**
- **Calculate Relative Branch Offsets and Perform Double Precision Hexadecimal Arithmetic**
- **Program 2716 or 2732 Type EPROMs**

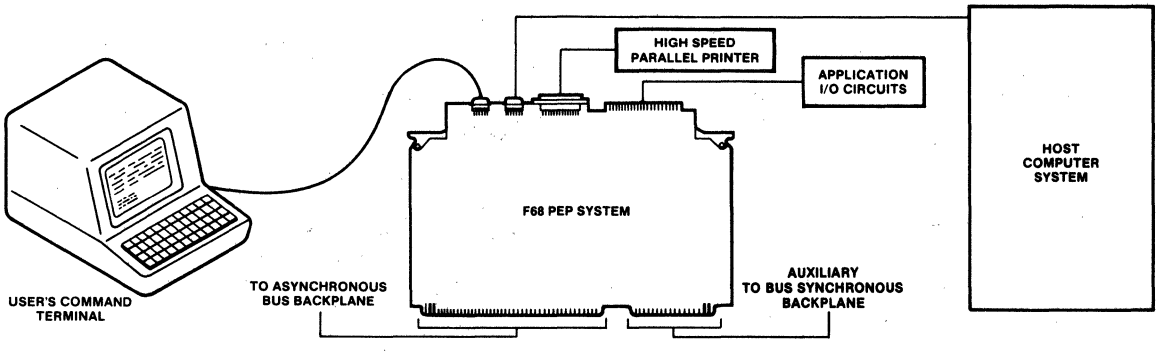
- 32 User-Definable Functions
- Examine/Alter I/O Port Bits
- Single-Step Program Execution Through NN Instructions of a Program
- Step Through Instructions Conditionally Until Specified Condition is Met
- Trace Through NN Instructions Displaying the CPU Registers After Each Instruction
- Trace Through Instructions Displaying CPU Registers After Each Occurence Until Specified Condition is Met
- Transparent Mode Operation for Conversing with a Host Computer from the Same Command Terminal
- Echo Incoming Data from Either of the Serial I/O Channels to the Parallel Printer Port
- Echo Monitor Output to Parallel Printer Port for Hardcopy of Monitor Output
- Enter ASCII Strings to Memory
- Print ASCII Strings from Memory
- Keyboard Test Mode

Applications

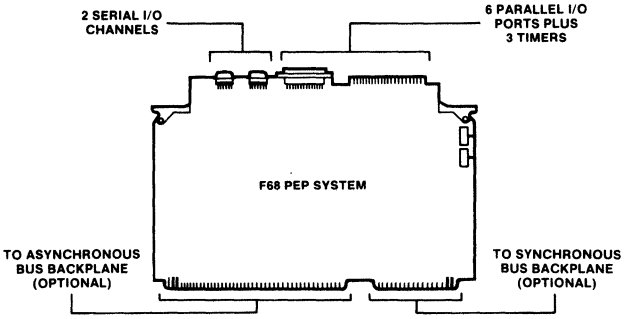
Low-Cost Development System



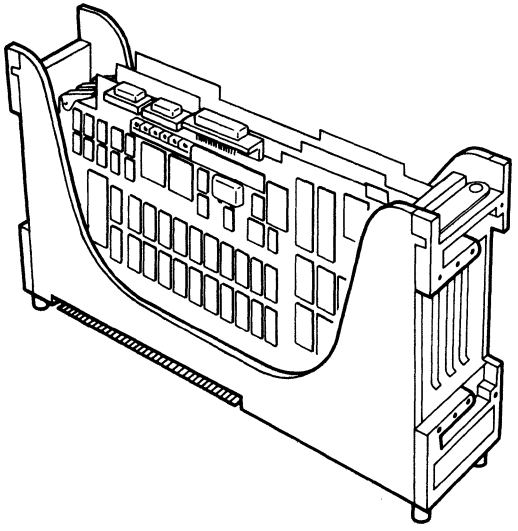
More Powerful Development System



Dedicated Use



Multi-Bus Card Cage Use



Hardware Specifications

Hardware Specifications

Microprocessor

CPU	6802, 6808, 6809
Data word size	8 bits (1 byte)
Instruction word size	6809: 1-4 bytes 6802/6808: 1-3 bytes
Cycle time	1.0 μs
System clock	4,000 MHz
Address capability	65,536 bytes

Memory

RAM	9K bytes, static 2114 RAM on-board
ROM	Six sockets for 24-pin ROMs or EPROMs. Accepts device types: 2516, 2716, 2532, 2732, 68316, 68332, 68364, or 68764 (i.e., anywhere from 2K to 48K bytes of ROM)
Expansion	External memory in any combination of RAM or ROM up to 64K bytes maximum

Input/Output

Parallel I/O	Six TTL-compatible, bidirectional 8-bit I/O ports with two port controls each
Serial I/O	Two programmable, asynchronous channels with full RS-232-C interfaces Each channel is double-buffered and has independent switch-selectable baud rates of 50, 110, 150, 300, 1200, 1800, 2400, 4800, 9600, or 19,200 bps

Timers

3 Binary Timers	Three separate 16-bit binary counters Each independently software controlable and readable Each with external clock and gate controls for frequency and pulse-width measurements Each with a counter output pin

Interrupts

Hardware	One non-maskable interrupt line available at both system bus edge connectors (wired-OR to the *Restart* pushbutton switch for initiating manual interrupts) One maskable interrupt line for fast interrupt response (6809 only) One maskable I/O interrupt line
Software	Software interrupts available: 1 for 6802/6808 3 for 6809

System Busses	
Dual Backplane Busses	P1—an 86-pin asynchronous system bus compatible with standard Multibus slave boards (multi-master options not supported) P2—a 60-pin synchronous MPU bus that allows complete expansion capabilities and faster operating speeds
I/O Busses	P3—a 60-pin applications bus with four parallel I/O ports with controls, plus three sets of counter controls for the three on-board binary timers P4—a 25-pin applications connector with two parallel I/O ports with controls; can be used for connection with a high-speed parallel printer (Centronics type) P5—a 9-pin RS-232-C serial I/O interface P6—a 9-pin RS-232-C serial I/O interface

Connectors	
	P1—86-contact, double-sided edge connector on 0.156" centers P2—60-contact, double-sided edge connector on 0.100" centers P3—60-contact, double-sided edge connector on 0.100" centers P4—25-pin, subminiature D-type right angle connector P5, P6—9-pin, subminiature D-type right angle connector

Power Supplies	
Requirements	+5 V_{dc} ±5% @ 2.5 A (typ) +25 V_{dc} @ 30 mA (typ) (used for EPROM programming only)

Environmental	
Temperature	0 to +50°C
Humidity	0 to 90% (noncondensing)

Physical Dimensions[2]	
Height	8.0 (203.2)
Length	12.0 (305)
Thickness	0.672 (17.1)
Weight	17 oz. (approximate)

Notes

1. Power can be applied to the board either through the card-edge backplane connector, or by connection of discrete wires to the on-board screw-down terminal strip.
2. All dimensions in inches bold and millimeters (parentheses)

9

Ordering Information

Order Number	Description
PEP680XCSD	PEP 68 Single Board Development System
	NN = without cross-assembler
	specifies CPU type 8 = 6802/6808 CPU or 9 = 6809 CPU
	for Intel MDS systems (single density floppy disk)
	for Intel MDS systems (double density floppy disk)

Relocatable Macro Cross-Assembler Software Packages for F6800, F6801, F6802, F6803 Using Intel MDS-800 Series Development Systems

The PEP 68 System is available with cross-assembler software packages that allow users of Intel development systems to do software development for F6800, F6801, F6802, and F6803 CPUs on their own systems. The cross-assembler software package is compatible with both the Motorola and Fairchild language syntax. Useful features similar to those of the 8080/8085 Assembler are included to provide systems compatibility.

The assembler accepts the user's source program as input and translates it into machine-executable code. Relocatable object modules are linked together into load modules and then into execution modules under the ISIS-II operating system. Application programs can then be downloaded in a formatted form through a serial port to the PEP 68 System's on-card memory. Now the program can be exercised and debugged using the FAIRBUG/68 debug monitor.

- **680X Cross-Assembler Software Package**
- **Intellec 800 and 888 Series II Compatible**
- **Full Macro Capability**
- **Expanded Relocation Capability**
- **Expanded Assembler Directives**
- **Comprehensive Conditional Assembly**
- **Includes Logical, Comparative, and Expression Truncation Operators**

Several software packages are offered by the Microprocessor Division for the system developer. A brief description of each one follows.

F8 Formulator Disk Operating System Version 4.0 (DOS4)

The Formulator Disk Operating System version 4.0 (DOS4) provides the F8 or F387X system developer with a complete set of tools for software development including source program editor, relocatable assembler, linking loader and interactive debugger. Also included are many utilities for efficient use of the floppy disk subsystem and support for a number of other standard peripherals. The DOS4 is an improved and streamlined version of the F8-D03 with added capability and greater ease of use.

FAST Software Debugger

The FAST software debugger (FSD) is a fast software debugging monitor for F8/F387X microcomputer systems programs. Its speed and ease of use meet or exceed any other method of debugging F8/F387X programs.

The FSD is designed for use with the Fairchild F387X programming, evaluation, and prototyping (PEP) board. It replaces the PEPBUB monitor chip provided with the board and allows all operations to be performed through a CRT terminal rather than through the PEGBUG keypad. It does not support parallel paper tape I/O.

FAIRBUG

A special Debug ROM 3851A PSU provides the F8 user with a convenient and powerful programming debug facility that is used in the development of F8 programs. This debugging program (FAIR-BUG) provides the user with an interactive system via a teletype terminal. The following capabilities are provided:

- Display or Alter Memory locations
- Display or Alter Scratchpad Registers
- Display of Alter Accumulator, ISAR,
  Status (W Register)
- Display or Alter PC0, DC0, DC1
- Load Formatted Paper Tape
- Punch Formatted Paper Tape
- Punch Paper Tape in PROM Format
- Entry from Keyboard or by Program Instruction
- I/O Subroutines available to user

F8/F3870 PEPBUG

A special F38T56 PSU with a debug monitor (PEPBUG 38) has been developed by Fairchild to provide the user with a convenient and powerful programming debug facility to aid in the development of F8 or F387X programs. The debugging program provides the user with an interactive system via a teletype terminal or via a 4 × 6 keypad. This is the standard debugging aid provided with the PEP 387X development board.

Minicomputer F8/F387X Cross Assembler

The Fairchild F8/F387X Cross Assembler is designed for use on any 16-bit word length minicomputer with an ANSI FORTRAN IV Compiler. The Cross Assembler is independent of machine character representation and numerical representation. Minor alterations may be required to satisfy various Computer/Operating System/Peripheral Device combinations.

Installation and modification of this program should be performed by a programmer who is quite familiar with FORTRAN IV and with the hardware and software configuration of the target computer. Under such circumstances, installation can probably be completed in one or two days.

F9445 BASIC Language Package

The Fairchild BASIC language interpreter for F9445-based systems is specifically tailored to high-performance microcomputing, providing a powerful, interactive programming language that can be used to solve a wide range of application problems. It incorporates extensions of and modifications to the BASIC language originally developed at Dartmouth College. The Fairchild enhanced BASIC increases the capability and flexibility of the language with a complete set of data types, additional statements and functions, comprehensive data management facilities, file management an I/O control and multi-dimensional array capabilities. Interface to custom F9445 assembly language programs is also provided. The BASIC language is fully supported by the F9445 Interactive Multi-User Disk Operating System (IMDOS), which allows full use of the extensive operating features of IMDOS, such as independent I/O and the ability to dynamically create, access, and delete files.

F9445 PEPBASIC Language Package

The Fairchild PEPBASIC, designed to reside in a 2K PROM, retains the essential simplicity and computational power of BASIC. PEPBASIC provides a unique capability to extend and customize programs, either through enhancements written by the user in F9445 absolute assembly language. Versatile applications like real-time process control, data acquisition, or math packages can be created, based on the general-purpose facilities available within PEPBASIC.

F9445 PEPBUG Package

The Fairchild F9445 PEPBUG package is the interactive entry and debugging software for use with the F9445 family of microprocessor products. The PEPBUG 45 software package creates a versatile and efficient control environment, enabling the user to enter and test F9445 absolute assembly language programs interactively. It is unique among the programs offered with the F9445 family in that it gives the user control of the microprcessor through a video terminal, provides many different capabilities in a single stand-alone mini-executive program, and occupies a relatively small amount (2 thousand bytes) of memory space.

F9445 PASCAL Language Package

The Fairchild F9445 PASCAL package is a high-level language suited to the development of microcomputer software because of its strong and logical control structures and its versatility in handling data. Fairchild PASCAL is designed to solve complex problems using such modern language concepts as variable data types, including records, sets, scalars, and others. Interface to custom F9445 assembly language subroutines is provided.

PASCAL offers highly structured techniques for organizing and coding programs so that they are easily understood and modified, which allows cost-effective software development.

F9445 Instruction Timer

The Fairchild F9445 Instruction Timer (TIMER45) software operates in the F9445-based systems, reporting the time needed to execute each class of CPU instruction. It uses the I/O terminal device as the standard to measure the times and report the results. The timer is most useful for detecting the execution speeds in mircroseconds for over 60 representative instructions, to optimize the design of F9445-based systems. It also serves as a diagnostic tool in detecting clock drift.

To time an instruction, a short loop containing the instruction is executed and its time lapse compared to a null (no instruction) loop, during the transmission of one character. The I/O terminal displays the resulting times. The user specifies the Baud rate of the I/O device at program execution time in reponse to a program prompt.

F9445 MICROFORTRAN Language Package

The Fairchild MICROFORTRAN package is a ligh-level language compatible with the F9445 microprocessor based family of products, providing a powerful tool for structured program development. Subroutines and functions are independently compiled, and translated into relocatable object modules that can be linked in any combination, according to commands given at load time. Interface to custom F9445 assembly language subroutines is provided.

F9445 Tests Package

The basic diagnostic package for the Fairchild F9445 family of microprocessor products contains seven programs: a memory address test, a memory test, a system exerciser, a memory diagostic, and three F9445 instruction set tests. These disk-based programs enable the user to identify and isolate faults in the CPU, memory, and certain I/O subsystems of F9445-based systems. Versions of several of the tests also test the Fairchild System-I (FS-I).

F9445 Interactive Multi-User Disk Operating System (IMDOS)

The Fairchild F9445 Interactive Multi-User Disk Operating System (IMDOS), customized for high-performance microcomputer systems, offers extended file management, timesharing, device-independent input/output, system processors such as MACRO assembler and a utility library. F9445 PASCAL, F9445 BASIC, and MICRO FORTRAN compilers are also fully supported. IMDOS is also the principal operating system for the Fairchild System-I (FS-I).

Section 10
Applications

FAIRCHILD
A Schlumberger Company

A Matrix Printer Controller Using The F8 and F3870 Circuits

The multi-chip F8™ microprocessor and single-chip F3870 MicroMachine™2 microcomputer have become popular circuits for control applications. Inexpensive and easy to use, their instruction sets and architecture combine to give the modern system designer NMOS LSI power and flexibility.

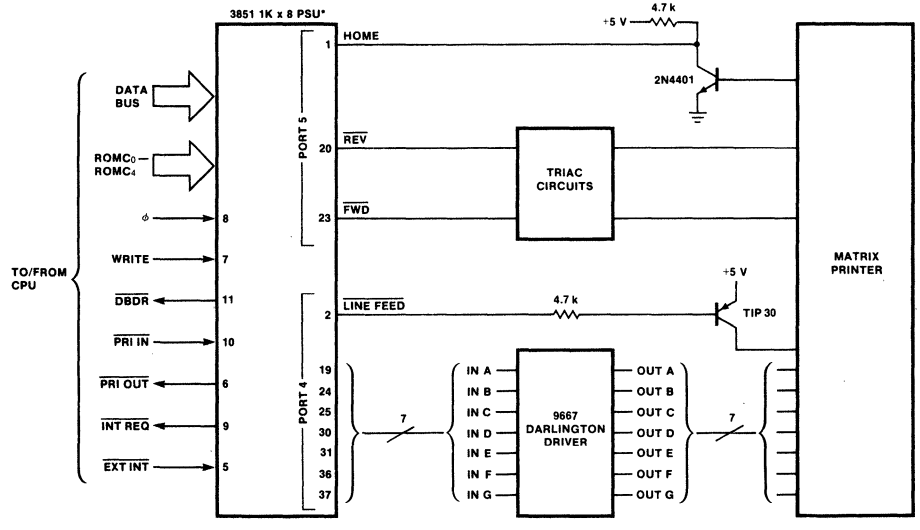
The architecture of the F8 microprocessor is designed to implement I/O-intensive applications. The memory addressing registers, the 16-bit program counter, and the data counters are located in the Program Storage Unit (PSU). The PSU, as well as the other F8-system peripheral circuits, is driven by the Central Processing Unit (CPU) with micro-instructions communicated over the five control lines ($ROMC_0$-$ROMC_4$) and is synchronized by a Write signal. The unusual partitioning of the CPU and PSU chips frees many pins normally needed for address bussing for use as I/O lines and provides room for a 64-byte scratchpad memory on the CPU chip. No matter how much memory is contained in the system, the number of I/O lines remains fixed at 16; therefore, the number of pins available for useful functions does not diminish as memory size increases.

The F8 microprocessor can address up to 64K bytes of program and data storage. Each peripheral controller can easily be implemented as a subroutine within the PSU and, depending upon the desired configuration, the required PSU can be plugged in to provide a modular, flexible system.

The F3870 MicoMachine2 is a complete 8-bit microcomputer on a single MOS integrated circuit. It features 2048 bytes of ROM, 64 bytes of scratchpad RAM, a programmable binary timer, 32 bits of I/O, and has a single +5 V power supply requirement. The F3870 can execute the F8 instruction set and can easily be interfaced with any microprocessor system through the I/O ports by properly defining command, status, and data lines, making it a universal controller.

MATRIX PRINTER CONTROLLER

A matrix printer controller can be constructed using either the F8 microprocessor *(Figure 1)* or the F3870 microcom-



*Other PSUs available are the 3856 (2K x 8 with I/O) and the 3857 (2K x 8 with address bus).

Fig. 1 Matrix Printer Controller Using the F8 Microprocessor

10

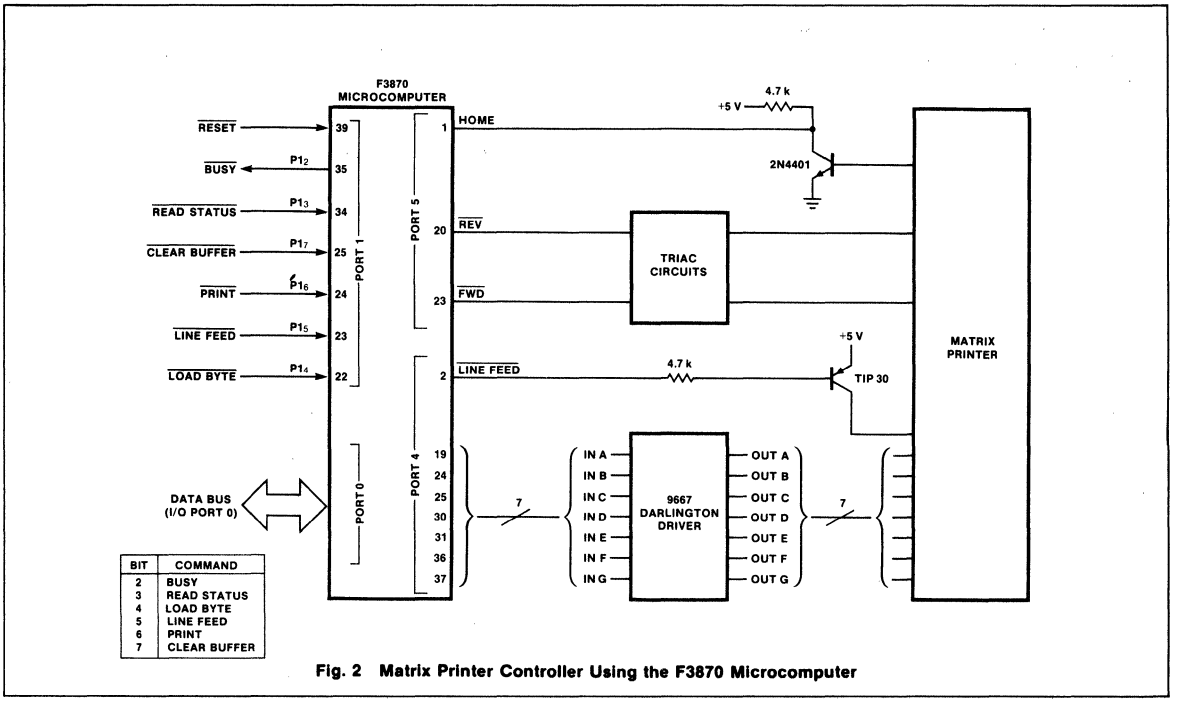


Fig. 2 Matrix Printer Controller Using the F3870 Microcomputer

puter *(Figure 2)*. In the F3870-based controller, the following commands are used to perform the control functions:

$\overline{\text{CLEAR}}$ $\overline{\text{BUFFER}}$ - stores zeroes in the 40-character print buffer contained within the scratchpad RAM.

$\overline{\text{PRINT}}$ - causes the contents of the print buffer to be printed. Error status if the head motion is not correct.

$\overline{\text{LINE}}$ $\overline{\text{FEED}}$ - advances the paper to the next line.

$\overline{\text{LOAD}}$ $\overline{\text{BYTE}}$ - takes a byte from the data bus and places it next in the print buffer. Error status if the buffer is full.

$\overline{\text{READ}}$ $\overline{\text{STATUS}}$ - places the status on the data bus and clears the status byte. The status is held on the bus until the command is taken away, at which time the port is cleared for reading again.

In all command sequences, the F3870 microcomputer presents $\overline{\text{BUSY}}$ until the command has been performed or until status is stable on the data bus.

The current requirements of the matrix printer solenoids are met by a suitable driver, such as the 9667 Darlington driver circuit with seven drivers and built-in back-emf suppression diodes. The 9667 interfaces directly with the F8 microprocessor and F3870 microcomputer ports.

The line-feed drive solenoid is implemented as a pnp power transistor (TIP 30), the base drive of which is supplied directly from the I/O port. The HOME phototransistor in the matrix printer supplies base current to a simple 2N4401 npn transistor, which saturates, providing the Home signal to the controller. The forward and reverse triac drives are provided across 100 Ω resistors from +5 V *(Figure 3)*. A TTL gate is used to hold the gate current off and provide a low-impedance path to ground. This provides good noise immunity to prevent turn-on of either triac by noise.

SOFTWARE DESIGN

The matrix printer controller software can easily fit within a 3851 PSU with 1K x 8 bits of ROM and 16 I/O lines.

The timing can be done by software loops without using the timer. This is the easiest technique, but suffers from the drawback that the whole system is tied up during the printing of an entire line. A more sophisticated technique, employed in many real-time control systems, is to make each timing control event a discrete event entered into a table controlled by the real-time monitor.

The software has three entry points:

The initialization entry point (address H'00B1'), which fires the reverse triac, turns off the paper-feed solenoid, and returns the print head to the home position.

The line-feed entry point (address H'00CF'), which energizes the paper-feed solenoid for 30 ms and then turns off the solenoid.

The print entry point (address H'0065'), which fires the forward triac, prints the line of characters, fires the reverse triac, and then does a paper feed.

Access to this software is accomplished by loading the registers with the required parameters and executing a "call to subroutine immediate" (PI) instruction to the appropriate entry point.

The subroutines to control the matrix printer head motion

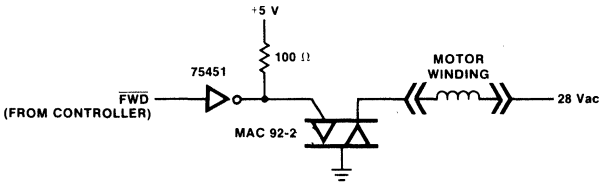


Fig. 3 Forward Triac Interface Circuit

and printing functions are listed in the appendix. These would be used alone in a 3851 PSU with other F8 system circuits or as part of an F3870 universal controller. The control program for the F3870 microcomputer and its subroutines are listed in the appendix.

FORWARD MOTION CONTROL

The forward triac is fired by setting bit 1 in output port 5:

```
LIS     2
OUTS    5
```

All other bits in port 5 should be cleared so that it is not necessary to OR bit 1 to the port. The Home signal is connected to port 5 bit 7 and active High (+5 = Home). This makes use of the fact that the F8 system input instructions also set the status register. A test for minus then detects when Home becomes false:

```
INS     5       INPUT & SET STATUS
BP      *-1     LOOP UNTIL "HOME" IS FALSE
```

However, it must be determined if the forward motion fails for some reason. Therefore, the system does not loop indefinitely but, rather, sets up two counters and waits only 1.5 second, see program segment A.

PRINT SOLENOID CONTROL

Once the Home indications goes false, the system fires the print solenoids, waits 650 μs, turns off the solenoids, and waits 700 μs for each of the five columns forming the character, see program segment B.

Segment A

```
        CLR
        LR      0,A
        LR      1,A
PRDR20  INS     5
        BM      PRDR30      } 24 μs x 256 = 5.9 ms
        DS      0           }
        BNZ     PRDR20      }
        DS      1           } 1.5 s
        BNZ     PRDR20      }
        CLR
        OUTS    5           TURN OFF FWD TRIAC
        LIS     2           SET ERROR FLAG
        LR      ERR,A
        BR      EXIT
```

Segment B

```
PRDR30  LM                  LOAD FIRING PATTERN
        OUTS    4           }
        LI      186         }
        INC                 } 9 μs x 70 = 630 μs
        BNZ     *-1         } 645 μs "ON TIME"
        CLR                 }
        OUTS    4           }
        LI      180         }
        INC                 } 9 μs x 76 = 684 μs
        BNZ     *-1         } 698 μs "OFF TIME"
        DS      EOC         }
        BNZ     PRDR30      }
```

REVERSE MOTION CONTROL
The forward triac is turned off and a 10-ms delay initiated to allow sufficient time for the triac to stop conduction (one-half cycle is 8.3 ms). The reverse triac is then fired and the program loops until Home becomes true. Again, there is some error control in the event that something prevents the print head from returning to the home position, see program segment C.

LINE FEED CONTROL
The line-feed solenoid can be turned on for only 30 ms; beyond that time, damage may be done to it. Setting bit 7 turns on the solenoid:

```
LI      H'80'
OUTS    4          TURN ON SOLENOID
LIS     10    }
LR      1,A   }
DS      0     }    30 ms DELAY
BNZ     *-1   }
DS      1     }
BNZ     *-4   }
CLR
OUTS    4          TURN OFF SOLENOID
```

CHARACTER SET TABLE
Accessing tables of data with the F8 microprocessor and F3870 MicroMachine 2 microcomputer is easy and efficient. The data counter is loaded using the "load dc immediate" (DCI) instruction. The "add accumulator to data counter" (ADC) instruction allows a signed 8-bit value contained in the accumulator to be added to the data counter. When the data is loaded from the table, the address is incremented by one, pointing to the next value in the table.

The table is organized so that the first bit pattern is addressed by pointing to the beginning of the table and adding the ASCII character to the data counter five times:

```
DCI     BIT PAT   TABLE ADDRESS
ADC       }       POINTS TO
ADC       }       THE Nth
ADC       }       ENTRY IN A
ADC       }       FIVE-BYTE-
ADC       }       WIDE TABLE
```

Since the first 32 ASCII characters are not used in this matrix printer, the actual program subtracts 32 from the ASCII character before adding it to the data counter five times.

CONCLUSION
The F8 instruction set has been shown to be ideal for control applications, such as the matrix printer controller described. Of particular note are the input/output instructions that set status, and the table look-up instructions that allow fast access to tables of any length and do not place any constraints on the location of tables in memory.

The F3870 microcomputer has been shown to be ideal for use with any microprocessor system as a universal peripheral controller. This is accomplished by interfacing through the input/output ports, which gives the system designer great flexibility in his system configuration.

Segment C

```
            CLR              TURN OFF FWD TRIAC
            OUTS   5
            LIS    3     }
            LR     1,A   }
            DS     0     }   10 ms DELAY
            BNZ    *-1   }
            DS     1     }
            BNZ    *-4   }
            LIS    1         TURN ON REVERSE TRIAC
            OUTS   5
            CLR              CLEAR COUNTERS FOR TIMEOUT DELAY
            LR     0,A
            LR     1,A
PRHO10      INS    5
            BP     PRHO20
            DS     0     }
            BNZ    PRDO10 }  1.5 s TIMEOUT
            DS     1     }
            BNZ    PRHO10 }
            LIS    1         SET ERROR STATUS
            LR     ERR,A
PRHO20      LIS    3     }
            LR     1,A   }
            DS     0     }   10 ms DELAY
            BNZ    *-1   }
            DS     1
            BNZ    *-4
            CLR              TURN OFF REVERSE TRIAC
            OUTS   5
```

```
 FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE              SOURCE STATEMENT

                    0001  *  MATRIX PRINTER CONTROLLER
                    0002  *
                    0003  *  D. R. HOLLINBECK
                    0004  *  FAIRCHILD MOS MICROCOMPUTER
                    0005  *
                    0006  *  THE UNIVERSAL CONTROLLER CONTROL
                    0007  *  PROGRAM IS GIVEN FIRST WITH THE
                    0008  *  SUBROUTINES AND BIT PATTERN TABLES
                    0009  *  FOLLOWING
                    0010  *
               0005 0011  STATUS   EQU   5          STATUS BYTE
               0004 0012  BYTES    EQU   4          NUMBER OF BYTES TO PRINT
               0003 0013  EOC      EQU   3          COUNTER FOR END-OF-CHARACTE

               0018 0014  BUFFER   EQU   D'100'-40 START OF BUFFER
               0004 0015  BUSY     EQU   B'00000100' 'BUSY' BIT
               00FB 0016  NBUSY    EQU   B'11111011' NOT 'BUSY' BIT
                    0017  *
     0000 70         0018          CLR             CLEAR ACCUMULATOR
     0001 B0         0019          OUTS  0         ALLOW READING OF PORTS
     0002 B1         0020          OUTS  1              0 AND 1.
     0003 2800B1 00B1 0021         PI    PRHOME      INSURE HEAD IS HOME
     0006 55         0022          LR    STATUS,A  CLEAR STATUS
     0007 54         0023          LR    BYTES,A     AND BYTE COUNTER
                     0024  *
                     0025  *  READ COMMAND STROBES
                     0026  *
     0008 A1         0027  RDCMD   INS   1
     0009 84FE   0008 0028         BZ    RDCMD       WAIT FOR COMMAND
     000B 913C   0048 0029         BM    CLRBUF      TEST BIT 7
     000D 13         0030          SL    1
     000E 914D   005C 0031         BM    PRINT       TEST BIT 6
     0010 13         0032          SL    1
     0011 9118   002A 0033         BM    LINEFD      TEST BIT 5
     0013 13         0034          SL    1
     0014 911E   0033 0035         BM    LDBYTE      TEST BIT 4
                     0036  *
                     0037  *  READ STATUS COMMAND
                     0038  *
     0016 45         0039          LR    A,STATUS  GET STATUS FROM SCRATCHPAD
     0017 B0         0040          OUTS  0
     0018 A1         0041          INS   1         SET 'BUSY'
     0019 2204       0042          OI    BUSY
     001B B1         0043          OUTS  1
     001C A1         0044          INS   1
     001D 2108       0045          NI    B'00001000'  WAIT FOR COMMAND
     001F 94FC   001C 0046         BNZ   *-3            TO GO AWAY
     0021 70         0047          CLR             CLEAR STATUS AND PORT
     0022 B0         0048          OUTS  0
     0023 55         0049          LR    STATUS,A
                     0050  *
                     0051  *  CLEAR BUSY STATUS
                     0052  *
     0024 A1         0053  CLRBSY  INS   1
     0025 21FB       0054          NI    NBUSY       RESET 'BUSY'
     0027 B1         0055          OUTS  1
     0028 90DF   0008 0056         BR    RDCMD       WAIT FOR ANOTHER COMMAND
                     0057  *
                     0058  *  LINE FEED COMMAND
                     0059  *
     002A A1         0060  LINEFD  INS   1              SET 'BUSY'
     002B 2204       0061          OI    BUSY
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE              SOURCE STATEMENT

    002D B1           0062            OUTS  1
     002E 2800CF 00CF 0063             PI    LF00        GO DO LINE FEED
     0031 90F2   0024 0064             BR    CLRBSY      CLEAR 'BUSY' AND WAIT
                      0065  *
                      0066  *  TRANSFER BYTE INTO PRINT BUFFER
                      0067  *
     0033 2019        0068  LDBYTE   LI    BUFFER      POINT TO NEXT EMPTY
     0035 C4          0069            AS    BYTES         BYTE IN BUFFER
     0036 2540        0070            CI    D'100'      CHECK IF BUFFER FULL
     0038 9407   0040 0071            BNZ   LDBY10
                      0072  *
                      0073  *  ERROR - SET BUFFER FULL STATUS (BIT 3)
                      0074  *     AND ERROR FLAG (BIT 7).
                      0075  *
     003A 45          0076            LR    A,STATUS    GET STATUS BYTE
     003B 2288        0077            OI    B'10001000'   SET ERROR FLAGS
     003D 55          0078            LR    STATUS,A
     003E 90E5   0024 0079            BR    CLRBSY
                      0080  *
     0040 0B          0081  LDBY10   LR    IS,A        LOAD ISAR
     0041 A0          0082            INS   0           GET DATA
     0042 5C          0083            LR    S,A         STORE INTO SCRATCHPAD
     0043 44          0084            LR    A,BYTES     INCREMENT COUNTER
     0044 1F          0085            INC
     0045 54          0086            LR    BYTES,A
     0046 90DD   0024 0087            BR    CLRBSY
                      0088  *
                      0089  *  CLEAR PRINT BUFFER COMMAND
                      0090  *
     0048 A1          0091  CLRBUF    INS   1
     0049 2204        0092            OI    BUSY        SET 'BUSY'
     004B B1          0093            OUTS  1
     004C 2028        0094            LI    40          40 BYTES TO CLEAR
     004E 54          0095            LR    BYTES,A
     004F 2019        0096            LI    BUFFER      GET STARTING ADDRESS
     0051 0B          0097            LR    IS,A          AND PUT INTO ISAR
     0052 70          0098  CLRB10   CLR               CLEAR ACCUMULATOR
     0053 5C          0099            LR    S,A         STORE VIA ISAR
     0054 0A          0100            LR    A,IS        INCREMENT ISAR
     0055 1F          0101            INC
     0056 0B          0102            LR    IS,A
     0057 34          0103            DS    BYTES
     0058 94F9   0052 0104            BNZ   CLRB10
     005A 90C9   0024 0105            BR    CLRBSY
                      0106  *
                      0107  *  PRINT BUFFER COMMAND
                      0108  *
     005C A1          0109  PRINT     INS   1
     005D 2204        0110            OI    BUSY
     005F B1          0111            OUTS  1
     0060 280065 0065 0112            PI    PRDR00
     0063 90C0   0024 0113            BR    CLRBSY
```

```
 FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE              SOURCE STATEMENT

                      0114            EJECT
                       0115  *
                       0116  *
                       0117  *  MATRIX PRINTER DRIVER
                       0118  *  D. R. HOLLINBECK
                       0119  *  FAIRCHILD MOS MICROCOMPUTER
                       0120  *
                       0121  *
                       0122  *  MAIN PRINT ENTRY POINT
                       0123  *
      0065 08          0124  PRDR00   LR    K,P        SAVE RETURN ADDRESS
      0066 11          0125           LR    H,DC       SAVE DC0
      0067 72          0126           LIS   2          FIRE FORWARD TRIAC
      0068 B5          0127           OUTS  5
      0069 10          0128  PRDR10   LR    DC,H       RESTORE LIST POINTER
      006A 16          0129           LM               GET NEXT CHARACTER TO PRINT
      006B 24E0        0130           AI    -32
      006D 11          0131           LR    H,DC       SAVE DC0
      006E 2A00E0 00E0 0132           DCI   BITPAT     POINTER TO PRINT TABLE
      0071 8E          0133           ADC              DC0 = DC0 + 5 * (ACC)
      0072 8E          0134           ADC
      0073 8E          0135           ADC              THIS GETS THE PROPER ENTRY
      0074 8E          0136           ADC              IN A FIVE BYTE WIDE TABLE.
      0075 8E          0137           ADC
      0076 70          0138           CLR
      0077 50          0139           LR    0,A        INITIALIZE DELAY COUNTER
      0078 51          0140           LR    1,A        INITIALIZE ERROR CODE
      0079 75          0141           LIS   5          LOAD NEEDLE FIRING COUNTER
      007A 53          0142           LR    EOC,A
      007B A5          0143  PRDR20   INS   5          LOOK AT 'HOME' LED INDICATO

      007C 910F   008C 0144           BM    PRDR30
      007E 30          0145           DS    0
      007F 94FB   007B 0146           BNZ   PRDR20
      0081 31          0147           DS    1
      0082 94F8   007B 0148           BNZ   PRDR20
      0084 70          0149           CLR
      0085 B5          0150           OUTS  5
      0086 45          0151           LR    A,STATUS
      0087 2282        0152           OI    B'10000010'
      0089 55          0153           LR    STATUS,A
      008A 9027   00B2 0154           BR    PRHD05     TURNOFF TRIAC'S AND EXIT
      008C 16          0155  PRDR30   LM               GET NEEDLE DRIVER BITS
      008D B4          0156           OUTS  4          FIRE DRIVERS
      008E 20BA        0157           LI    186        WAIT 650 MICROSECONDS
      0090 1F          0158           INC
      0091 94FE   0090 0159           BNZ   *-1
      0093 70          0160           CLR              RESET NEEDLE DRIVERS
      0094 B4          0161           OUTS  4
      0095 20B4        0162           LI    180        DELAY 700 MICROSECS
      0097 1F          0163           INC
      0098 94FE   0097 0164           BNZ   *-1
      009A 33          0165           DS    EOC        TEST END-OF-CHARACTER
      009B 94F0   008C 0166           BNZ   PRDR30     GET NEXT BIT PATTERN
      009D 2074        0167           LI    116        DELAY 1350 MICROSECS
      009F 1F          0168           INC
      00A0 94FE   009F 0169           BNZ   *-1
      00A2 34          0170           DS    BYTES      TEST FOR END-OF-LINE
      00A3 94C5   0069 0171           BNZ   PRDR10     GET NEXT BYTE TO PRINT
      00A5 70          0172           CLR              TURN OFF FORWARD TRIAC
      00A6 B5          0173           OUTS  5
      00A7 73          0174           LIS   3          DELAY 10 MILLISECS
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE              SOURCE STATEMENT

    00A8 51           0175          LR    1,A
     00A9 30          0176          DS    0
     00AA 94FE   00A9 0177          BNZ   *-1
     00AC 31          0178          DS    1
     00AD 94FB   00A9 0179          BNZ   *-4
     00AF 9002   00B2 0180          BR    PRH005
                      0181  *
                      0182  *  INITIALIZATION ENTRY POINT.
                      0183  *     ENTER HERE AT THE START OF THE MAIN PROGRAM
                      0184  *     JUST TO ENSURE THAT THE PRINT HEAD IS HOME.
                      0185  *
     00B1 08          0186  PRHOME  LR    K,P         SAVE RETURN ADDRESS
                      0187  *
                      0188  *  ENTER HERE TO RETURN PRINT HEAD HOME
                      0189  *
     00B2 71          0190  PRH005  LIS   1           FIRE REVERSE TRIAC
     00B3 B5          0191          OUTS  5
     00B4 70          0192          CLR
     00B5 50          0193          LR    0,A
     00B6 51          0194          LR    1,A
     00B7 A5          0195  PRH010  INS   5           WAIT FOR 'HOME'
     00B8 810B   00C4 0196          BP    PRH020
     00BA 30          0197          DS    0
     00BB 94FB   00B7 0198          BNZ   PRH010
     00BD 31          0199          DS    1
     00BE 94F8   00B7 0200          BNZ   PRH010
     00C0 45          0201          LR    A,STATUS
     00C1 2281        0202          OI    B'10000001' HEAD DID NOT RETURN HOME
     00C3 55          0203          LR    STATUS,A
     00C4 73          0204  PRH020  LIS   3           DELAY 10 MILLISECS
     00C5 51          0205          LR    1,A
     00C6 30          0206          DS    0
     00C7 94FE   00C6 0207          BNZ   *-1
     00C9 31          0208          DS    1
     00CA 94FB   00C6 0209          BNZ   *-4
     00CC 70          0210          CLR               TURN OFF
     00CD B5          0211          OUTS  5             REV TRIAC
     00CE 0C          0212          PK                RETURN
                      0213  *
                      0214  *  LINE FEED ENTRY POINT
                      0215  *
     00CF 08          0216  LF00    LR    K,P
     00D0 70          0217          CLR               TURN OFF TRIAC'S
     00D1 B5          0218          OUTS  5
     00D2 2080        0219          LI    H'80'       FIRE LINE FEED
     00D4 B4          0220          OUTS  4
     00D5 7A          0221          LIS   10          DELAY 30 MILLISECS
     00D6 51          0222          LR    1,A
     00D7 30          0223          DS    0
     00D8 94FE   00D7 0224          BNZ   *-1
     00DA 31          0225          DS    1
     00DB 94FB   00D7 0226          BNZ   *-4
     00DD 70          0227          CLR               TURN OFF LINE FEED
     00DE B4          0228          OUTS  4
     00DF 0C          0229          PK                RETURN FROM SUBROUTINE
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE              SOURCE STATEMENT

                    0230          EJECT
                    0231  *
                    0232  *  TABLE OF BIT PATTERNS FOR CHARACTER SET.
                    0233  *
    00E0 00         0234  BITPAT    DC    H'00'
    00E1 00         0235          DC    H'00'
    00E2 00         0236          DC    H'00'                          ( )
    00E3 00         0237          DC    H'00'
    00E4 00         0238          DC    H'00'
                    0239  *
    00E5 00         0240          DC    H'00'
    00E6 00         0241          DC    H'00'
    00E7 7D         0242          DC    H'7D'          *****  *       (!)
    00E8 00         0243          DC    H'00'
    00E9 00         0244          DC    H'00'
                    0245  *
    00EA 00         0246          DC    H'00'
    00EB 60         0247          DC    H'60'          **
    00EC 00         0248          DC    H'00'                          (")
    00ED 60         0249          DC    H'60'          **
    00EE 00         0250          DC    H'00'
                    0251  *
    00EF 14         0252          DC    H'14'            * *
    00F0 7F         0253          DC    H'7F'          *******
    00F1 14         0254          DC    H'14'            * *
    00F2 7F         0255          DC    H'7F'          *******
    00F3 14         0256          DC    H'14'            * *
                    0257  *
    00F4 12         0258          DC    H'12'            *  *
    00F5 2A         0259          DC    H'2A'           * * *
    00F6 7F         0260          DC    H'7F'          *******         ($)
    00F7 2A         0261          DC    H'2A'           * * *
    00F8 24         0262          DC    H'24'           *  *
                    0263  *
    00F9 62         0264          DC    H'62'          **   *
    00FA 64         0265          DC    H'64'          **  *
    00FB 08         0266          DC    H'08'             *            (%)
    00FC 13         0267          DC    H'13'            *  **
    00FD 23         0268          DC    H'23'           *   **
                    0269  *
    00FE 36         0270          DC    H'36'           ** **
    00FF 49         0271          DC    H'49'          *  *  *
    0100 35         0272          DC    H'35'           ** * *         (&)
    0101 02         0273          DC    H'02'              *
    0102 05         0274          DC    H'05'             * *
                    0275  *
    0103 00         0276          DC    H'00'
    0104 68         0277          DC    H'68'          ** *
    0105 70         0278          DC    H'70'          ***            (')
    0106 00         0279          DC    H'00'
    0107 00         0280          DC    H'00'
                    0231  *
    0108 00         0282          DC    H'00'
    0109 1C         0283          DC    H'1C'            ***
    010A 24         0284          DC    H'24'           *   *          (()
    010B 42         0285          DC    H'42'          *     *
    010C 00         0286          DC    H'00'
                    0287  *
    010D 00         0288          DC    H'00'
    010E 42         0289          DC    H'42'          *     *
    010F 24         0290          DC    H'24'           *   *          ())
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE                SOURCE STATEMENT

    0110 1C          0291           DC    H'1C'
     0111 00          0292           DC    H'00'
                      0293  *
     0112 08          0294           DC    H'08'
     0113 2A          0295           DC    H'2A'
     0114 1C          0296           DC    H'1C'          (*)
     0115 2A          0297           DC    H'2A'
     0116 08          0298           DC    H'08'
                      0299  *
     0117 08          0300           DC    H'08'
     0118 08          0301           DC    H'08'
     0119 7F          0302           DC    H'7F'          (+)
     011A 08          0303           DC    H'08'
     011B 08          0304           DC    H'08'
                      0305  *
     011C 00          0306           DC    H'00'
     011D 0D          0307           DC    H'0D'
     011E 0E          0308           DC    H'0E'          (,)
     011F 00          0309           DC    H'00'
     0120 00          0310           DC    H'00'
                      0311  *
     0121 08          0312           DC    H'08'
     0122 08          0313           DC    H'08'
     0123 08          0314           DC    H'08'          (-)
     0124 08          0315           DC    H'08'
     0125 08          0316           DC    H'08'
                      0317  *
     0126 00          0318           DC    H'00'
     0127 03          0319           DC    H'03'
     0128 03          0320           DC    H'03'          (.)
     0129 00          0321           DC    H'00'
     012A 00          0322           DC    H'00'
                      0323  *
     012B 02          0324           DC    H'02'
     012C 04          0325           DC    H'04'
     012D 08          0326           DC    H'08'          (/)
     012E 10          0327           DC    H'10'
     012F 20          0328           DC    H'20'
                      0329  *
     0130 3E          0330           DC    H'3E'
     0131 45          0331           DC    H'45'
     0132 49          0332           DC    H'49'          (0)
     0133 51          0333           DC    H'51'
     0134 3E          0334           DC    H'3E'
                      0335  *
     0135 00          0336           DC    H'00'
     0136 21          0337           DC    H'21'
     0137 7F          0338           DC    H'7F'          (1)
     0138 01          0339           DC    H'01'
     0139 00          0340           DC    H'00'
                      0341  *
     013A 23          0342           DC    H'23'
     013B 45          0343           DC    H'45'
     013C 49          0344           DC    H'49'          (2)
     013D 49          0345           DC    H'49'
     013E 31          0346           DC    H'31'
                      0347  *
     013F 22          0348           DC    H'22'
     0140 41          0349           DC    H'41'
     0141 49          0350           DC    H'49'          (3)
     0142 49          0351           DC    H'49'
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE                SOURCE STATEMENT

    0143 36          0352          DC    H'36'          ** **
                     0353  *
     0144 0C         0354          DC    H'0C'            **
     0145 14         0355          DC    H'14'           * *
     0146 24         0356          DC    H'24'          *  *     (4)
     0147 7F         0357          DC    H'7F'          *******
     0148 04         0358          DC    H'04'            *
                     0359  *
     0149 72         0360          DC    H'72'          ***  *
     014A 51         0361          DC    H'51'          * *   *
     014B 51         0362          DC    H'51'          * *   *   (5)
     014C 51         0363          DC    H'51'          * *   *
     014D 4E         0364          DC    H'4E'          *  ***
                     0365  *
     014E 1E         0366          DC    H'1E'           ****
     014F 29         0367          DC    H'29'          * *  *
     0150 49         0368          DC    H'49'          *  *  *   (6)
     0151 49         0369          DC    H'49'          *  *  *
     0152 46         0370          DC    H'46'          *   **
                     0371  *
     0153 40         0372          DC    H'40'          *
     0154 47         0373          DC    H'47'          *   ***
     0155 48         0374          DC    H'48'          *  *      (7)
     0156 50         0375          DC    H'50'          * *
     0157 60         0376          DC    H'60'          **
                     0377  *
     0158 36         0378          DC    H'36'           ** **
     0159 49         0379          DC    H'49'          *  *  *
     015A 49         0380          DC    H'49'          *  *  *   (8)
     015B 49         0381          DC    H'49'          *  *  *
     015C 36         0382          DC    H'36'           ** **
                     0383  *
     015D 30         0384          DC    H'30'           **
     015E 49         0385          DC    H'49'          *  *  *
     015F 49         0386          DC    H'49'          *  *  *   (9)
     0160 4A         0387          DC    H'4A'          *  * *
     0161 3C         0388          DC    H'3C'           ****
                     0389  *
     0162 00         0390          DC    H'00'
     0163 36         0391          DC    H'36'          ** **
     0164 36         0392          DC    H'36'          ** **     (:)
     0165 00         0393          DC    H'00'
     0166 00         0394          DC    H'00'
                     0395  *
     0167 00         0396          DC    H'00'
     0168 6D         0397          DC    H'6D'          ** ** *
     0169 6E         0398          DC    H'6E'          ** ***    (;)
     016A 00         0399          DC    H'00'
     016B 00         0400          DC    H'00'
                     0401  *
     016C 08         0402          DC    H'08'            *
     016D 14         0403          DC    H'14'           * *
     016E 22         0404          DC    H'22'          *   *     (<)
     016F 41         0405          DC    H'41'          *     *
     0170 00         0406          DC    H'00'
                     0407  *
     0171 14         0408          DC    H'14'           * *
     0172 14         0409          DC    H'14'           * *
     0173 14         0410          DC    H'14'           * *     (=)
     0174 14         0411          DC    H'14'           * *
     0175 14         0412          DC    H'14'           * *
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE                SOURCE STATEMENT

                    0413  *
     0176 00        0414          DC    H'00'
     0177 41        0415          DC    H'41'          *     *
     0178 22        0416          DC    H'22'           *   *       (>)
     0179 14        0417          DC    H'14'            * *
     017A 08        0418          DC    H'08'             *
                    0419  *
     017B 30        0420          DC    H'30'           **
     017C 40        0421          DC    H'40'          *
     017D 45        0422          DC    H'45'          *   *  *     (?)
     017E 48        0423          DC    H'48'          *  *
     017F 30        0424          DC    H'30'           **
                    0425  *
     0180 3E        0426          DC    H'3E'           *****
     0181 41        0427          DC    H'41'          *     *
     0182 5D        0428          DC    H'5D'          * ***  *     (@)
     0183 55        0429          DC    H'55'          * * * *
     0184 3C        0430          DC    H'3C'           ****
                    0431  *
     0185 3F        0432          DC    H'3F'           ******
     0186 48        0433          DC    H'48'          *  *
     0187 48        0434          DC    H'48'          *  *         (A)
     0188 48        0435          DC    H'48'          *  *
     0189 3F        0436          DC    H'3F'           ******
                    0437  *
     018A 7F        0438          DC    H'7F'          *******
     018B 49        0439          DC    H'49'          *  *  *
     018C 49        0440          DC    H'49'          *  *  *     (B)
     018D 49        0441          DC    H'49'          *  *  *
     018E 36        0442          DC    H'36'           ** **
                    0443  *
     018F 3E        0444          DC    H'3E'           *****
     0190 41        0445          DC    H'41'          *     *
     0191 41        0446          DC    H'41'          *     *     (C)
     0192 41        0447          DC    H'41'          *     *
     0193 22        0448          DC    H'22'           *   *
                    0449  *
     0194 41        0450          DC    H'41'          *     *
     0195 7F        0451          DC    H'7F'          *******
     0196 41        0452          DC    H'41'          *     *     (D)
     0197 41        0453          DC    H'41'          *     *
     0198 3E        0454          DC    H'3E'           *****
                    0455  *
     0199 7F        0456          DC    H'7F'          *******
     019A 49        0457          DC    H'49'          *  *  *
     019B 49        0458          DC    H'49'          *  *  *     (E)
     019C 49        0459          DC    H'49'          *  *  *
     019D 41        0460          DC    H'41'          *     *
                    0461  *
     019E 7F        0462          DC    H'7F'          *******
     019F 48        0463          DC    H'48'          *  *
     01A0 48        0464          DC    H'48'          *  *         (F)
     01A1 48        0465          DC    H'48'          *  *
     01A2 40        0466          DC    H'40'          *
                    0467  *
     01A3 3E        0468          DC    H'3E'           *****
     01A4 41        0469          DC    H'41'          *     *
     01A5 41        0470          DC    H'41'          *     *     (G)
     01A6 45        0471          DC    H'45'          *   * *
     01A7 47        0472          DC    H'47'          *   ***
                    0473  *
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE                SOURCE STATEMENT

    01A8 7F          0474           DC    H'7F'          *******
     01A9 08         0475           DC    H'08'             *
     01AA 08         0476           DC    H'08'             *       (H)
     01AB 08         0477           DC    H'08'             *
     01AC 7F         0478           DC    H'7F'          *******
                     0479  *
     01AD 00         0480           DC    H'00'
     01AE 41         0481           DC    H'41'          *     *
     01AF 7F         0482           DC    H'7F'          *******       (I)
     01B0 41         0483           DC    H'41'          *     *
     01B1 00         0484           DC    H'00'
                     0485  *
     01B2 02         0486           DC    H'02'                *
     01B3 01         0487           DC    H'01'                 *
     01B4 01         0488           DC    H'01'                 *       (J)
     01B5 01         0489           DC    H'01'                 *
     01B6 7E         0490           DC    H'7E'          ******
                     0491  *
     01B7 7F         0492           DC    H'7F'          *******
     01B8 08         0493           DC    H'08'             *
     01B9 14         0494           DC    H'14'            * *          (K)
     01BA 22         0495           DC    H'22'           *   *
     01BB 41         0496           DC    H'41'          *     *
                     0497  *
     01BC 7F         0498           DC    H'7F'          *******
     01BD 01         0499           DC    H'01'                 *
     01BE 01         0500           DC    H'01'                 *       (L)
     01BF 01         0501           DC    H'01'                 *
     01C0 01         0502           DC    H'01'                 *
                     0503  *
     01C1 7F         0504           DC    H'7F'          *******
     01C2 20         0505           DC    H'20'             *
     01C3 18         0506           DC    H'18'              **         (M)
     01C4 20         0507           DC    H'20'             *
     01C5 7F         0508           DC    H'7F'          *******
                     0509  *
     01C6 7F         0510           DC    H'7F'          *******
     01C7 10         0511           DC    H'10'             *
     01C8 08         0512           DC    H'08'             *       (N)
     01C9 04         0513           DC    H'04'              *
     01CA 7F         0514           DC    H'7F'          *******
                     0515  *
     01CB 3E         0516           DC    H'3E'           *****
     01CC 41         0517           DC    H'41'          *     *
     01CD 41         0518           DC    H'41'          *     *       (O)
     01CE 41         0519           DC    H'41'          *     *
     01CF 3E         0520           DC    H'3E'           *****
                     0521  *
     01D0 7F         0522           DC    H'7F'          *******
     01D1 48         0523           DC    H'48'          *  *
     01D2 48         0524           DC    H'48'          *  *          (P)
     01D3 48         0525           DC    H'48'          *  *
     01D4 30         0526           DC    H'30'           **
                     0527  *
     01D5 3E         0528           DC    H'3E'           *****
     01D6 41         0529           DC    H'41'          *     *
     01D7 45         0530           DC    H'45'          *   * *       (Q)
     01D8 42         0531           DC    H'42'          *    *
     01D9 3D         0532           DC    H'3D'           **** *
                     0533  *
     01DA 7F         0534           DC    H'7F'          *******
```

```
FORMULATOR FDOS ASSEMBLER (REV 2.0)

RS  LOC OBJECT ADDR LINE              SOURCE STATEMENT

    01DB 48          0535             BC    H'48'          ◆  ◆
     01DC 4C         0536              DC    H'4C'          ◆  ◆◆      (R)
     01DD 4A         0537              DC    H'4A'          ◆  ◆ ◆
     01DE 31         0538              DC    H'31'           ◆◆   ◆
                     0539  ◆
     01DF 32         0540              DC    H'32'           ◆◆  ◆
     01E0 49         0541              DC    H'49'          ◆  ◆  ◆
     01E1 49         0542              DC    H'49'          ◆  ◆  ◆    (S)
     01E2 49         0543              DC    H'49'          ◆  ◆  ◆
     01E3 26         0544              DC    H'26'           ◆  ◆◆
                     0545  ◆
     01E4 40         0546              DC    H'40'          ◆
     01E5 40         0547              DC    H'40'          ◆
     01E6 7F         0548              DC    H'7F'          ◆◆◆◆◆◆◆    (T)
     01E7 40         0549              DC    H'40'          ◆
     01E8 40         0550              DC    H'40'          ◆
                     0551  ◆
     01E9 7E         0552              DC    H'7E'          ◆◆◆◆◆◆
     01EA 01         0553              DC    H'01'                 ◆
     01EB 01         0554              DC    H'01'                 ◆    (U)
     01EC 01         0555              DC    H'01'                 ◆
     01ED 7E         0556              DC    H'7E'          ◆◆◆◆◆◆
                     0557  ◆
     01EE 70         0558              DC    H'70'          ◆◆◆
     01EF 0C         0559              DC    H'0C'             ◆◆
     01F0 03         0560              DC    H'03'                ◆◆    (V)
     01F1 0C         0561              DC    H'0C'             ◆◆
     01F2 70         0562              DC    H'70'          ◆◆◆
                     0563  ◆
     01F3 7F         0564              DC    H'7F'          ◆◆◆◆◆◆◆
     01F4 02         0565              DC    H'02'                ◆
     01F5 0C         0566              DC    H'0C'             ◆◆      (W)
     01F6 02         0567              DC    H'02'                ◆
     01F7 7F         0568              DC    H'7F'          ◆◆◆◆◆◆◆
                     0569  ◆
     01F8 63         0570              DC    H'63'          ◆◆   ◆◆
     01F9 14         0571              DC    H'14'            ◆ ◆
     01FA 08         0572              DC    H'08'             ◆       (X)
     01FB 14         0573              DC    H'14'            ◆ ◆
     01FC 63         0574              DC    H'63'          ◆◆   ◆◆
                     0575  ◆
     01FD 60         0576              DC    H'60'          ◆◆
     01FE 10         0577              DC    H'10'            ◆
     01FF 0F         0578              DC    H'0F'               ◆◆◆◆    (Y)
     0200 10         0579              DC    H'10'            ◆
     0201 60         0580              DC    H'60'          ◆◆
                     0581  ◆
     0202 43         0582              DC    H'43'          ◆     ◆◆
     0203 45         0583              DC    H'45'          ◆   ◆ ◆
     0204 49         0584              DC    H'49'          ◆  ◆  ◆    (Z)
     0205 51         0585              DC    H'51'          ◆ ◆   ◆
     0206 61         0586              DC    H'61'          ◆◆    ◆
                     0587  ◆
                     0588              END
ERRS
```

FAIRCHILD
A Schlumberger Company

Microprocessor-Controlled Phase-Locked Loop Tuning System

The TV channel changer, the volume control adjustment and other controls with knob and dial-type readouts are taken for granted today, and simplified so that even a child can operate them. The present forms are the result of many years of evolution, from "tweeking of the Cat's Whisker" in the early days of crystal radio, to the two-handed operation required for separate tuning of radio frequency and detector stages in rf receivers, or tuning with sliding inductor coils. These all have evolved into the systems used today and have undergone considerable "human engineering." With the event of fully electronic controlled systems with calculator-type key boards, touch panels, remote control, etc., a similar evolution is taking place in tuning and display systems.

The various controls and displays for TV or AM/FM receiver tuning, volume, brightness, etc., require either direct or indirect human interface (*Figures 1* and *2*). They can take many forms, from the old familiar knob system with dial readout that is controlled by a switch, potentiometer or variable capacitor, to a remote control keyboard system or touch-control panel. The latter requires electronic control of the major functions such as tuning and time readout. Processing and controlling these various functions separately, using hardware, is inefficient, inflexible and costly. The obvious solution is to incorporate a central processing unit within the receiver to process the human interface controls, determine the function to be performed, and establish the manner in which the function is accomplished (*Figure 3*). Since different manufacturers have different receiver requirements, the central processing unit will differ from manufacturer to manufacturer, or even model to model.

The F3870 low-cost one-chip microprocessor with 2K bytes of on-board ROM, a 64-bit scratchpad RAM, and 32 bits (four 8-bit bytes) of TTL-compatible input/output is an ideal candidate for a central processor. The F3870 requires no peripheral devices except a crystal and power supplies. Using this processor, the type of control operation is programmed into the ROM at the factory. With efficient software programming, two or three programs could be stored in the ROM so that the receiver manufacturer may offer the same chassis with different features and operations.

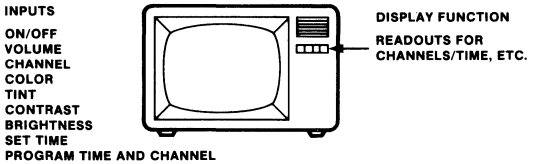


Fig. 1 Television

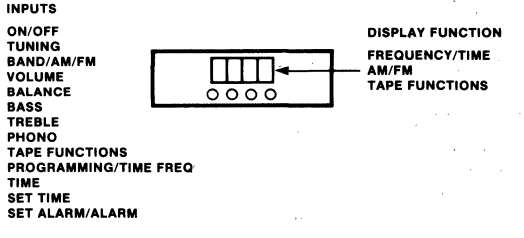


Fig. 2 Auto Radio/HI-FI

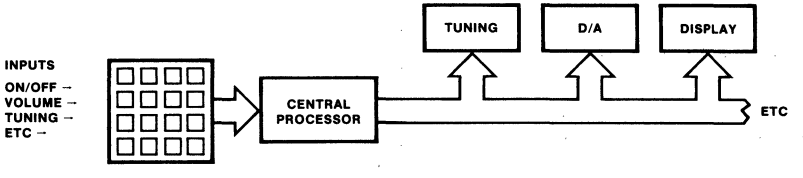


Fig. 3 Receiver with Central Processor

10

The microprocessor may also perform additional functions such as D/A conversion for actually controlling the circuits, tuning, volume, brightness, color, etc. It could be used as a time clock capable of being programmed to switch on the receiver at a given time and channel. The microprocessor may also act as a receiver section of a remote control system, ultrasonic or IR, to decode the signals for a particular function. However, specialized chips are now available that perform the dedicated functions—PLL timing, D/A conversion, etc.—more efficiently than microprocessors, and work well under microprocessor command.

STANDARDIZED BUS SYSTEMS

Fairchild adopted a standardized bus system (*Figure 4*) for microprocessor-controlled AM/FM and TV receivers. Using this system, AM/FM and TV models, from low end to the top of the line, may be built with almost every imaginable feature by simply adding the appropriate modular circuit onto the bus (*Figures 5* and *6*). Other circuits will be available shortly—an 8-channel 6-bit D/A converter, a time clock, and an on-screen character generator. Each modular chip will have built-in identity code, which is something like a chip select but operates on data on the data bus. The identity code word is four bits; therefore, there are 16 possible combinations, but only 15 are available for use since one is reserved for when no chip is addressed.

To address a particular chip on the data bus, the correct IDENT code is placed on the data bus and an IDENT clock on the control bus. The chip selected is initialized and reset, ready to accept data from the data bus. The DATA clock on the control bus clocks the information into the selected chip. The number

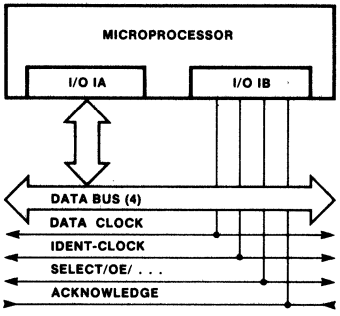


Fig. 4 Standardized Bus System

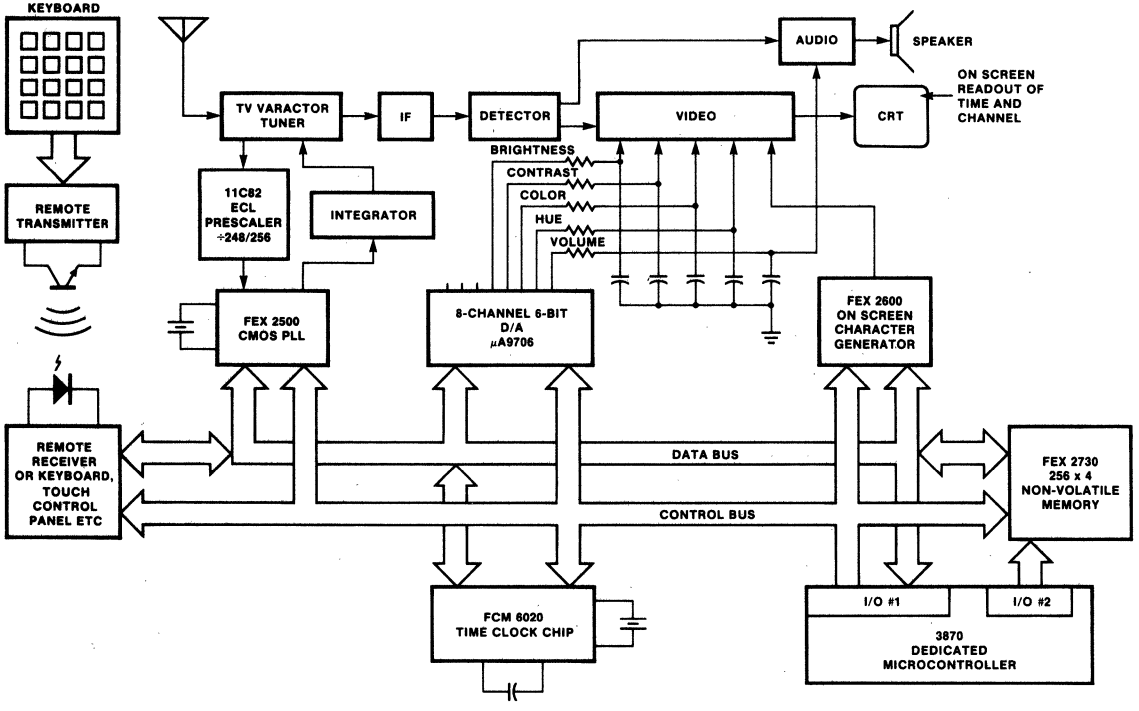


Fig. 5 Microprocessor-Controlled TV with PLL Tuning

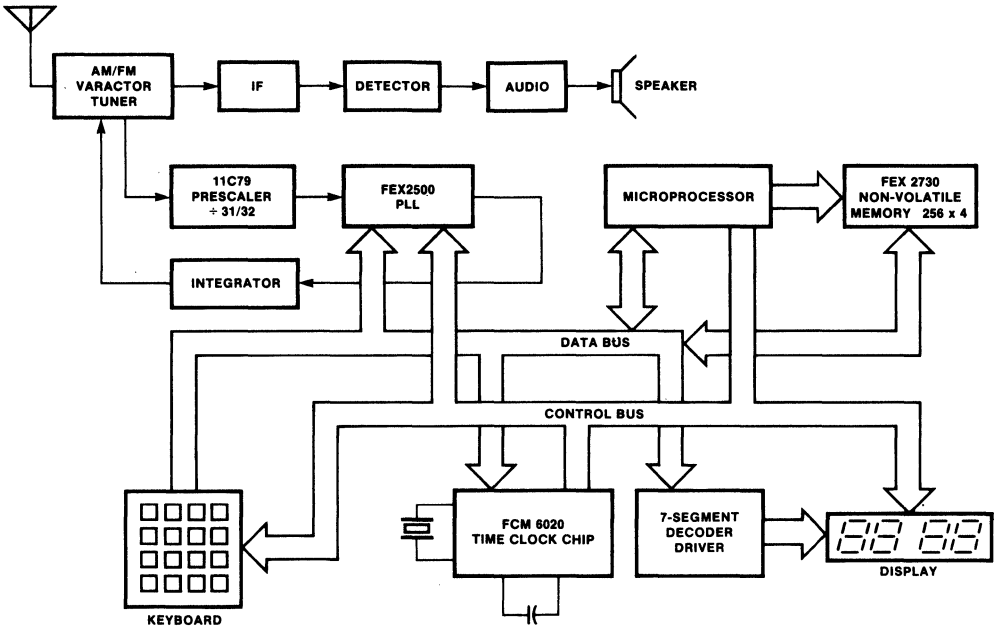


Fig. 6 Microprocessor-Controlled AM/FM Radio with PLL Tuning

of words required to read a chip depends upon the particular chip function, and a DATA clock must be generated for each word. To disable the chip from the bus, the procedure is reversed. The wrong IDENT code is put in the data bus with an IDENT clock on the control bus.

PLL TUNING

It has long been realized that the ultimate tuning system is the phase-locked loop. However, it has not successfully penetrated the consumer market, until recently, due to its stringent requirements—large complex logic, ability to perform at high frequencies, system partitioning, and specific system configuration. Simple PLL TV tuning systems accurately tuned to the FCC channel assignments; however, local problems such as antenna mismatch, IF misalignment, cable TV problems, etc., were often present and required fine tuning or, more accurately, detuning of the PLL system. Therefore, it was imperative to add the fine-tuning capability, inherent in the old turret tuners, to the PLL TV tuning system. This complicated the system and also required some kind of non-volatile memory for storing the fine-tuning information for each channel.

Another problem in PLL TV tuning was how to control the TV receiver, especially in the method of entry from the calculator-board—whether one or two keys should be used, or an entry button, and whether the channel display should indicate which key has been depressed or indicate the channel on the screen.

Now, with the addition of the microprocessor as a central control unit, the problems of the PLL TV tuning can be simplified and the exact application can be determined by the TV or AM/FM manufacturer rather than by the semiconductor manufacturer.

BASIC PLL SYSTEMS

A better, more descriptive name for the PLL would be "frequency, phase-locked loop" (*Figure 7*) because, for the loop to lock, the frequency must be adjusted first, then the phase.

The output frequency of the VCO (a varactor local oscillator) is divided down by the divide-by-n counter and fed into the frequency/phase detector where the frequency and phase are compared to a reference frequency. The frequency/phase detector output circuit has three modes—open circuit, or supplying a series of pump-up or pump-down pulse charges to the integrator/amplifier. The output of the integrator/amplifier supplies dc feedback control voltage to the VCO. When the loop is locked and operating at the desired frequency, the two frequencies fed into the frequency/phase detector are the same and of essentially the same phase.

Under these conditions, the frequency/phase comparator supplies only sufficient charge to maintain loop lock. When the loop is not locked, the two frequencies at the input to the frequency/phase comparator differ. The frequency/phase detector supplies a charge of sufficient amplitude and direction to the integrator/amplifier to generate a voltage for driving the VCO to a frequency that will cause the loop to lock. Therefore, the output frequency of the PLL can be written as follows:

$$f_{VCO} = n \times f_{REF}$$

where n is always a whole integer.

Changing the value of the divide-by-n counter changes the frequency. For example, a tuning system for tuning in 25 kHz increments requires a reference frequency of 25 kHz.

Unfortunately, to build a programmable divide-by-n counter for a PLL system that operates at the local oscillator frequencies of FM and TV receivers is impractical. Therefore, an ECL high-speed prescaler is inserted between the VCO and the programmable divide-by-n counter to reduce the counter frequency. A simple fixed prescaler (*Figure 8*) places too many compromises on the PLL designer—long loop lock-up times or

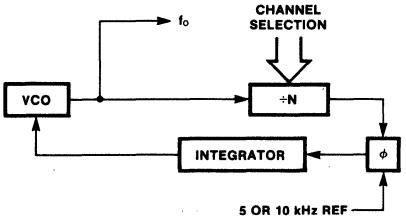


Fig. 7 Basic PLL Circuit

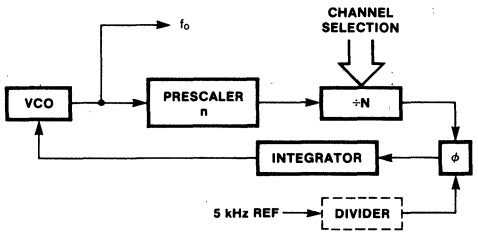


Fig. 8 High-Frequency Loop Using Fixed Prescaler

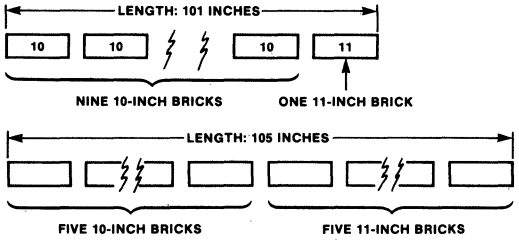


Fig. 9 Brick Analogy

restrictions on the timing increments—that generally result in too many loop or receiver problems. A superior approach is to use a dual-modulus prescaler, i.e., an ECL divider with two different divide ratios usually closely related, in a technique called "pulse swallowing."

Pulse Swallowing

Pulse swallowing is a technique that combines the talent of a very fast, but dumb, ECL prescaler with that of a low speed, but very smart, counter. The best way to explain it is to divert, for the moment, from the field of electronics and enter the world of masonry. Suppose a universal brick were required for building walls of any length, within one inch, without breaking the brick. One way to do this would be to make one brick 10 inches long and another 11 inches. Using combinations of these two sizes of bricks, walls of any length (over 100") may be built (*Figure 9*).

Back to electronics—the dual-modulus prescaler (*Figure 10*) is similar to the two different brick lengths. By controlling the dual-modulus prescaler appropriately, the incoming clock frequency pulses can be counted in two different "block lengths" (*Figure 11*). For high-frequency applications, pulse swallowing combines the advantages of both the straightforward method (*Figure 7*) and the fixed modulus prescaler (*Figure 8*). It allows the highest possible reference frequency f_{REF} and vastly reduces the speed requirement of a programmable divide-by-n counter. The dual-modulus prescaler, when operating, appears to swallow pulses when changing between the two divide ratios of the prescaler—thus the name, "pulse swallowing."

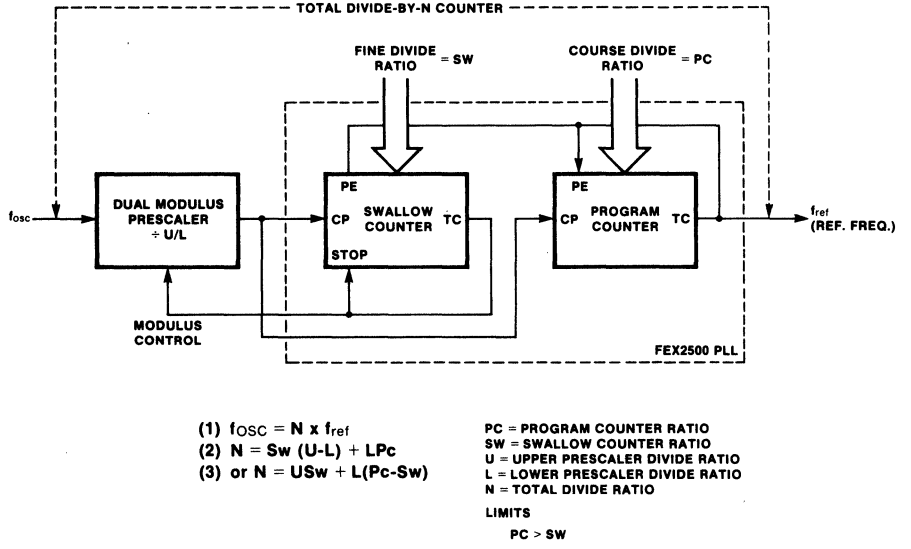
To keep track of how many times the prescaler operates in one of its two modes (usually the higher), an extra counter, called a swallow counter, is added to the system. The swallow counter has only a small total divide ratio compared to that of the program counter. It differs from the program counter in that its Terminal-Count output is connected back to a Stop input and operates like a one-shot. This is called a dead-ended counter because it stops after reaching terminal count.

The operation of the system for a total divide-by-n cycle is as follows. When the program counter reaches terminal count, it parallel loads both itself and the swallow counter with the desired divide ratios. The Terminal-Count output of the swallow counter, which is also the prescaler Mode-Control input, then goes HIGH and both counters begin to count. Since the swallow counter is smaller, it reaches terminal count first and stops, causing the Mode-Control input to the prescaler to go LOW, changing the prescaler modulus. The program counter continues until reaching terminal count and the divide-by-n sequence is repeated. Therefore the program counter performs the coarse, or rough, tuning while the swallow counter handles the fine tuning.

THE FAIRCHILD MICROPROCESSOR CONTROLLED PLL TUNING SYSTEM

The Fairchild FEX2500 PLL circuit is made using CMOS metal-gate technology and is packaged in a 28-pin DIP, either plastic or ceramic. The primary features are:

- Microprocessor addressable
- Data-holding registers independent of input data, once addressed
- Operates to 4 MHz
- Operates to 1 GHz with appropriate prescaler
- Fine tuning capability—1 kHz AM band, 25 kHz FM band, 62.5 kHz TV
- Complete digital portion of PLL tuning system
- On-board oscillator circuit for reference frequency
- 4 MHz, 2 MHz and 1 MHz outputs that may be used for clock input to microprocessor or other circuits
- Unique data-bus chip select system
- Choice of 1 kHz, 5 kHz, 7.8125 kHz or 25 kHz reference frequency
- Phase comparator incorporates patented anti-backlash circuit to reduce random FM modulation of the V_{CO}
- Out-of-lock output indicates out-of-lock condition of loop
- Dual ratio program and swallow counters for extended loop frequency range



(1) $f_{OSC} = N \times f_{ref}$
(2) $N = Sw\ (U-L) + LPc$
(3) or $N = USw + L(Pc-Sw)$

PC = PROGRAM COUNTER RATIO
SW = SWALLOW COUNTER RATIO
U = UPPER PRESCALER DIVIDE RATIO
L = LOWER PRESCALER DIVIDE RATIO
N = TOTAL DIVIDE RATIO

LIMITS

PC > SW

Fig. 10 Frequency Divide-by-N Counter Using a Dual Modulus Prescaler

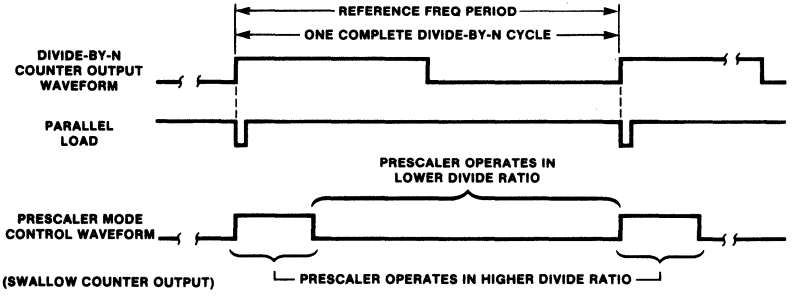


Fig. 11 Different "Block Lengths"

The FEX2500 contains all the essential digital components of an advanced PLL tuning system, requiring only an external tuner, integrator and a crystal to complete the entire loop.

The PLL loop operates from 100 kHz to 4 MHz directly, and to 1 GHz with the addition of an ECL prescaler. The system has fine tuning increments of 1 kHz for the AM band, 25 kHz for the FM band, and 62.5 kHz for TV and can be set lower with loop compromises. The user has a choice of 1 kHz, 5 kHz, 7.8125 kHz or 25 kHz reference frequencies when used with a 4 MHz reference crystal. Once addressed to the desired tuning frequency, the PLL circuit can be operated independently of the microprocessor, since the PLL contains the necessary holding registers for data. To accommodate the 4 MHz input operating frequency and low propagation delays, it uses two power supplies, +12 V and 5 V. The +5 V supply is required to make the input and output circuits compatible with other +5 V logic circuits. *Figure 12* shows the complete block diagram containing both the program counter and swallow counter as well as a phase comparator, crystal oscillator, reference frequency divider, four 4-bit registers to store tuning information, plus additional circuits.

Programs and Swallow Counters

Clock Inputs

The Clock input for the program and swallow counters can be selected for either a differential input or a single-ended input by using an Input Select pin (IS). For TV or FM radio applications, the differential input mode is generally used. The PLL differential inputs are connected directly to the differential outputs of an ECL prescaler. This considerably reduces the amount of radiated digital-interference noise. Single-ended mode is intended for AM radio applications without the use of a prescaler. In AM/FM radio, the local oscillator is connected to the single-ended input while the differential inputs are connected to the ECL prescaler that derives input from the FM local oscillator. Loop change from AM to FM is accomplished with the IS control. The maximum frequency on both inputs is 4 MHz.

Counter Configurations

The counter has a total bit length of 16 bits, is sub-divided into two sections—a program counter and a swallow counter, and has two selectable configurations: 13 bits program/3 bits swallow or 11 bits program/5 bits swallow. Counter-configuration control determines which bit lengths are selected according to

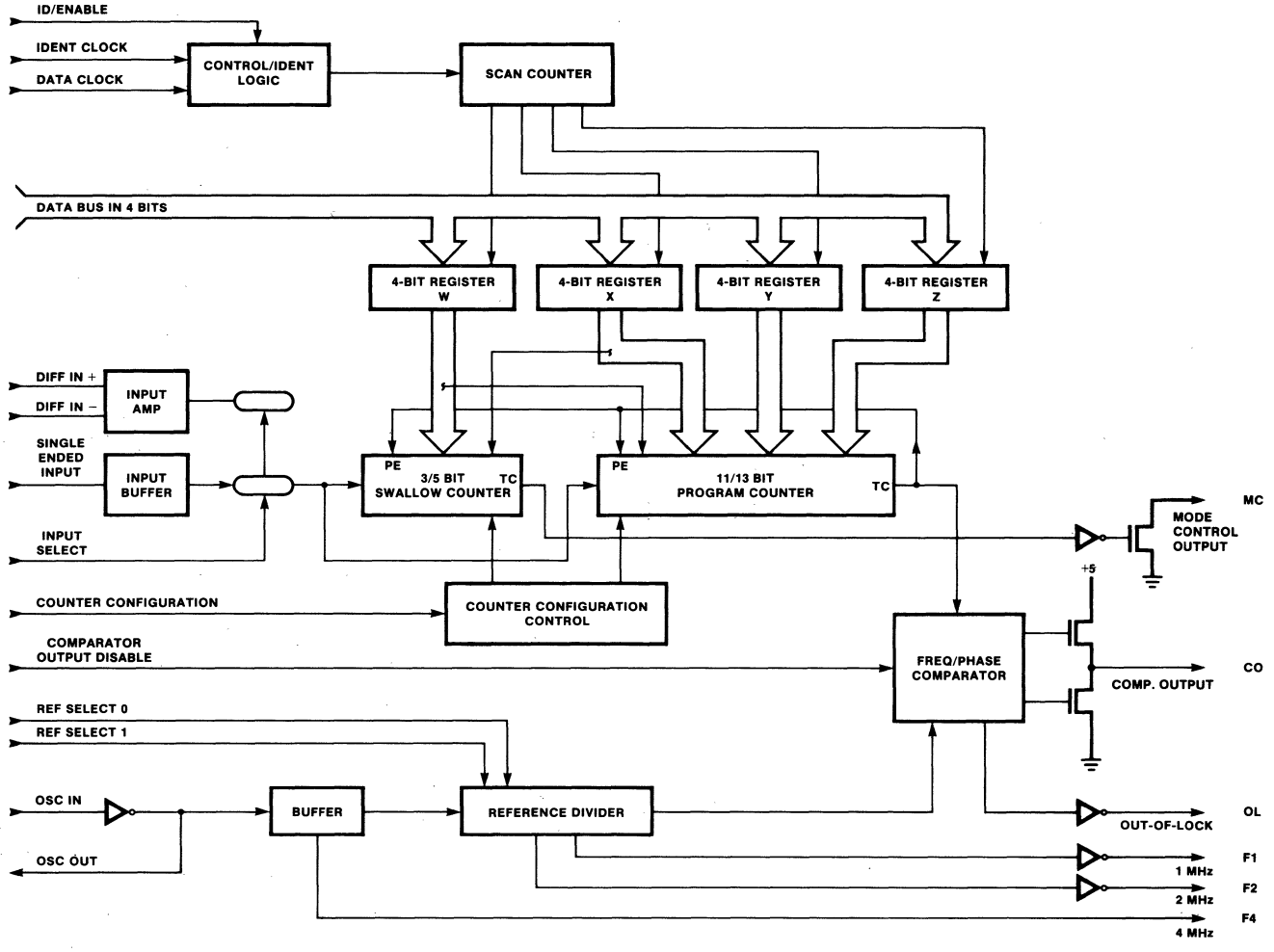


Fig. 12 FEX2500 PLL Block Diagram

different divide ratios required in various applications. The maximum possible divide ratios are:

MODE	PROGRAM	SWALLOW
13/3	$2^{13} = 8192$	$2^3 = 8$
11/5	$2^{11} = 2048$	$2^5 = 32$

Both counters are down types; therefore, the divide ratio is the same as that of the binary load value. The load values for frequency or channel allocation are obtained from the microprocessor. Binary counters, rather than decimal counters, simplify chip design and minimize software programming problems on the microprocessor. For low-frequency operation below 4 MHz, the swallow counter is not used and it is immaterial what data values are loaded into it. *Figure 13* shows the data loading format.

Frequency/Phase Comparator

The frequency/phase comparator has a number of unique features, an anti-backlash circuit, an out-of-lock detector and an output disable circuit. The actual frequency/phase comparator is the standard digital type that locks onto the negative edges of the two waveforms, one from the program counter and the other from the reference-frequency divider (*Figure 14*). It can only lock onto the correct frequency with no output of the comparator at multiples of either of the two frequencies.

Anti-Backlash Circuit

The anti-backlash circuit eliminates the dead-zone problems due to propagation delays in other digital frequency/phase comparators. A narrow pulse (≈ 200 ns) is injected into the pump-down circuit to cause a loop error. The loop responds by making another pulse of equal and opposite magnitude to cancel out the error (*Figure 15*). Both pulses are arranged to be closely related in time so they can be easily filtered out by small filter capacitors. Therefore the net charge fed to the integrator by these pulses is zero. However, the injection of these pulses causes a slight phase error that operates the frequency-phase comparator outside the dead zone. The addition of this circuit considerably enhances the spectral output of the VCO, eliminating random low-frequency modulation caused by phase-comparator "hunting" in standard comparators.

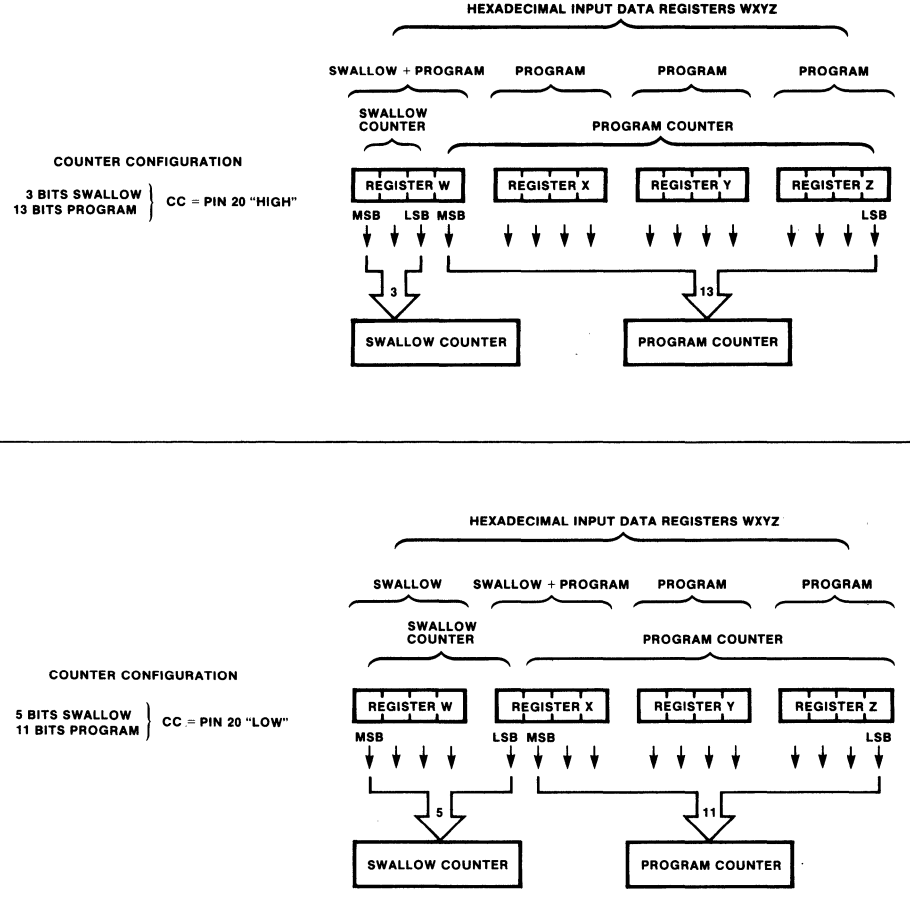


Fig. 13 Bit Coordination Between Input Registers and Loop Counters

10

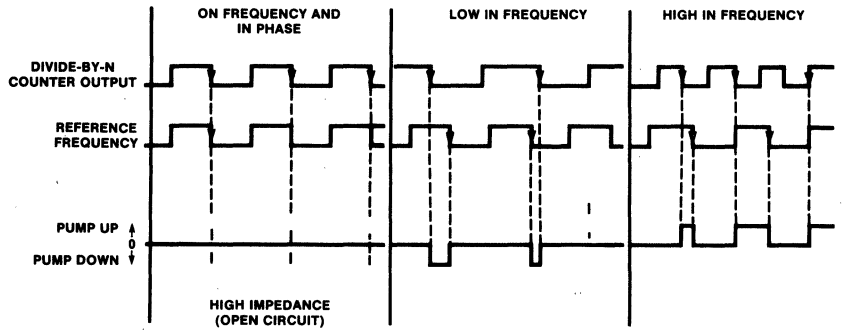


Fig. 14 Output Waveforms from Frequency/Phase Comparator

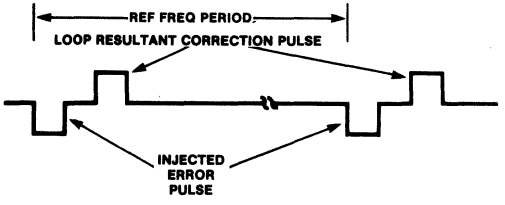


Fig. 15 Output of Frequency/Phase Detector with Anti-Backlash Circuit

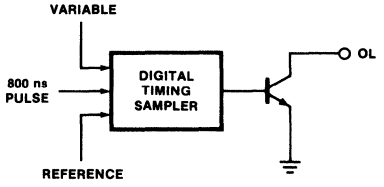


Fig. 16 Out-of-Lock Circuit

Out-of-Lock Detector (*Figure 16*)
This circuit is used to detect an out-of-lock condition, due to either a malfunction or a channel change. The indicator is also useful during initial loop set-up and test. It can also be used to mute the audio during channel changes, or it can be connected to an LED display for indication of loop malfunction.

In operation, a window signal is decoded off the reference-frequency divider chain (*Figure 17*). When the loop is locked, the negative edge of the output waveform from the divide-by-n counter rests in the middle of the window. If it strays outside the window for two period reference cycles, a latch is set and an out-of-lock condition is indicated.

Output Circuit
The output circuit is designed to provide three output modes (*Figure 18*)—current sourcing (pump-up), current sinking (pump-down) or high impedance (open circuit).

When the loop is locked, the output circuit is in the high-impedance state, except during the positive and negative anti-backlash pulse injection mentioned previously.

Output Disable
This control can be used to hold the loop on frequency while new frequency information is being loaded into the PLL registers. When the output is disabled, the output from the frequency phase comparator is in the high impedance state. When a frequency change is initiated, new data is fed, relatively slowly, into the four registers, WXYZ. During the loading of each register, the loop responds to the new data causing erroneous loop responses, unless the comparator output is disabled.

When this output is disabled, the integrator tends to remember the last charge level, thus keeping the loop on or about frequency.

Reference Frequency Divider and Oscillator Circuits
The oscillator circuit consists of two high-gain CMOS inverter circuits in series plus the external components—crystal, trimmer capacitor, resistor—connected between the input and output of the inverters.

The reference frequency divider circuit is a straightforward counter with various outputs—4 MHz, 2 MHz and 1 MHz—which may be used for the microprocessor clock. The reference frequencies of 1 kHz, 5 kHz, 7.8125 kHz and 25 kHz are tapped off the divider chain through a 4-input multiplexer, that selects the reference frequency.

Chip-Identity Circuit
When the IDENT code (PLL Chip 0110) is present on the Data Bus input and an IDENT clock is generated, the scan counter is initialized. Data can now be entered into the first register W via the DATA clock, which then clocks the scan counter so the second register X is ready to accept data. This is repeated until the remainder of the registers are loaded. Data can be repeatedly loaded into the registers using the DATA clock, but care must be exercised to keep track of which register is receiving data. To disable the Data Bus input to the PLL chip, a wrong IDENT code is put into the data bus and an IDENT clock generated. The IDENT code circuit may be disabled by leaving the ID Enable input HIGH.

Data Input Terminal
Figure 13 shows the input data format for registers W X Y Z in both configurations of the counters.

The generation of the input data for operating a PLL at given frequencies is worked out from the equation in *Figure 10*. The following example shows the procedure for establishing the

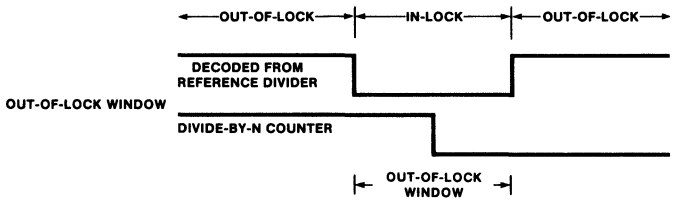


Fig. 17 Out-of-Lock Timing Diagram

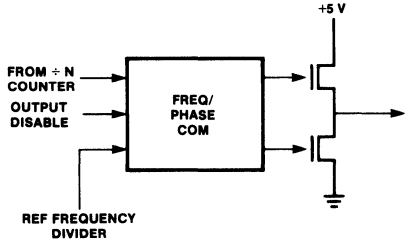


Fig. 18 Frequency/Phase Comparator Output

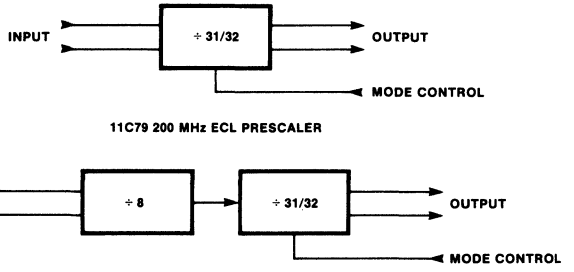


Fig. 19 Prescalers

values and will help clarify the operation of the pulse swallowing system.

Receiver -FM 88.1 to 107.9 MHz
IF 10.7 MHz

PLL Setup for 4 MHz Crystal Reference
25 kHz Reference Frequency

1. LOW Frequency = Receiver + IF
   = 88.1 + 10.7
   = 98.8 MHz

2. Total Divide-by-n = LOW Frequency ÷ Reference
   = 98.8 ÷ 25 kHz
   = 3952

3. To find PLL loading data:

   Program Counter = PC
   Swallow Counter = SW

   a. Find PC divide ratio (Coarse Tuning)

   "Divide-by-n" divided by prescaler lower ratio

   3952 ÷ 31 = 127.48387

   The integer part of the number is the PC load divide ratio

   ∴ PC = 127

   b. Find SW divide ratio (Fine Tuning)

   Take the integer number from above and multiply by prescaler lower ratio:

   127 x 31 = 3939

   Subtract this number from Divide-by-n:

   3952 - 3937 = 15

   ∴ SW = 15

   This is the number of times the prescaler operates in the divide-by-32 mode. It is not necessary to work out subsequent frequencies in the range, but merely increment the values by the required amount. These numbers then have to be translated first into binary, then into hexadecimal data to fit the data loading format shown in *Figure 13*.

PRESCALERS (*Figure 19*)
Two prescalers have been specially designed to operate with the FEX2500 PLL chip: 11C79 divide-by 31/32 (200 MHz) and

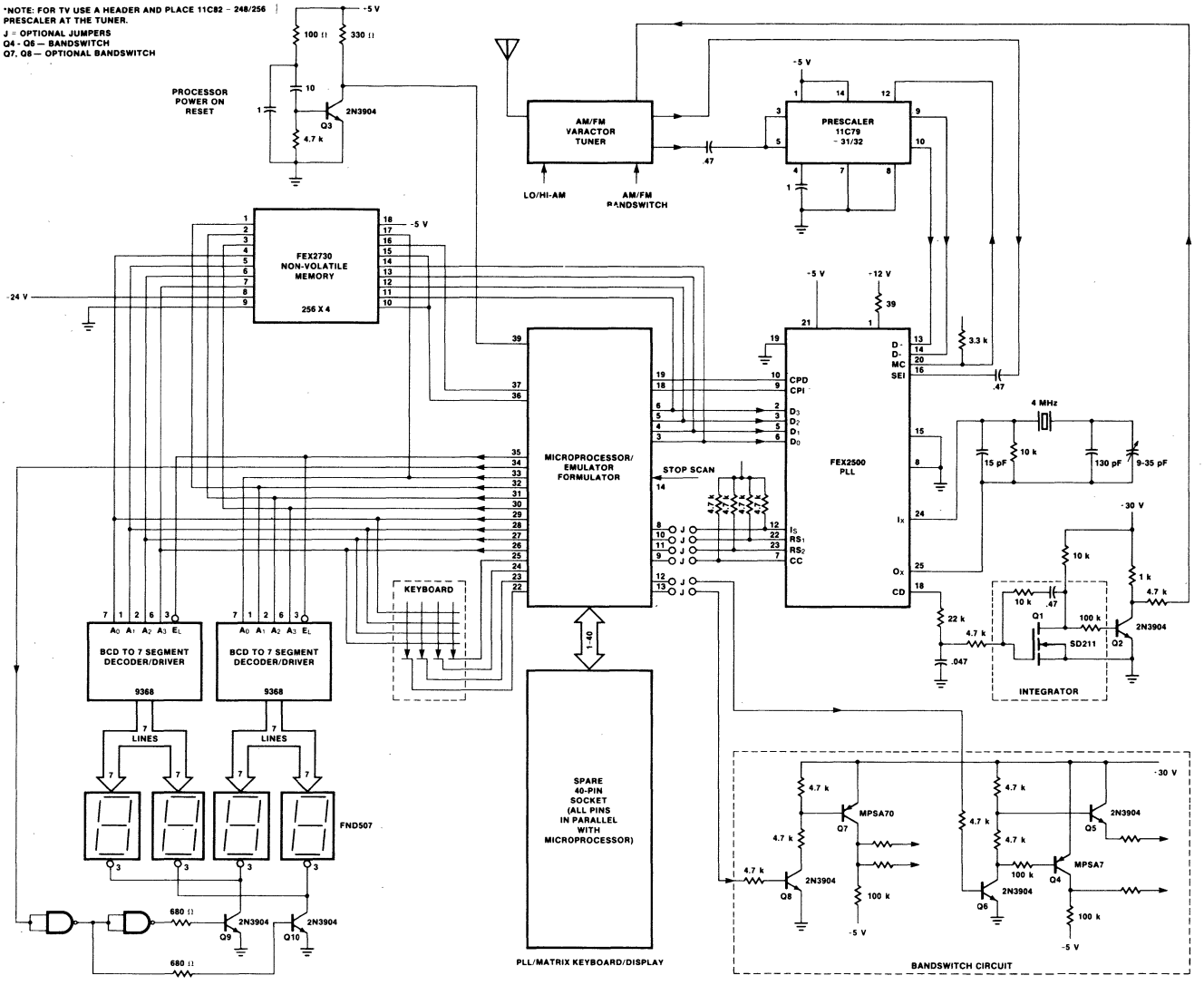


Fig. 20 Advanced AM/FM Radio Application

11C82 divide-by 248/25 (1 GHz). Other prescalers are available with a decrease in digital noise performance: 11C90 divide-by 10/11 (600 MHz) and 11C91 divide-by 5/6 (600 MHz).

The maximum operating frequency of PLL system is limited by the 4 MHz maximum frequency of PLL FEX2500.

Example using 11C90 divide-by 10/11:

= 4 MHz x 10 = 40 MHz

AM/FM APPLICATIONS

Figure 20 shows an advanced AM/FM radio with a single chip F3870 microprocessor and keyboard with an LED display read-out. It has manual or automatic search tuning and storage for a number of stations. It can tune on AM to 1 kHz increments and FM to 25 kHz.

With the integrator circuit, the undesirable reference frequency modulation of the VCO frequency is:

FM = better than 50 dB at 25 kHz
AM = better than 45 dB at 1 kHz

One exciting area for a PLL tuning system is a home AM/FM radio with short-wave bands. Low-cost short-wave receivers to date are generally difficult to tune but with the precise tuning of a PLL tuning circuit, this would change.

If the radio had memory storage for a number of short-wave stations, people living in different lands could program in a station from their own country. Most foreign countries radiate the same program material on a number of frequencies. Due to periodic sunspot activity, some frequencies are better than others; thus it is necessary to store a number of short-wave frequencies. Also, because of the ease of tuning, the short-wave broadcast bands could become acceptable for normal broadcast frequencies.

HISTORICAL NOTE ON PULSE SWALLOWING

The pulse swallowing technique makes the whole PLL tuning system commercially feasible and acceptable in performance. It was the idea, or invention, of John Nichols in about 1968-69 time period when he worked for Fairchild. The work was done initially for 360-channel aircraft radio/transmitters. It was not until many years later that his brilliant idea became widely known.

FAIRCHILD
A Schlumberger Company

Microprocessor-Based Solar Controller

Microprocessor-Based Solar Controller

Energy is being consumed today in greater quantities than ever; at the same time, yesterday's seemingly unlimited resources are now seen to be quite finite. As a result, energy conservation has assumed a new importance, and the search for alternative energy sources has begun in earnest. One of the more promising possibilities is harnessing the sun as a direct source of heat.

The solar heating systems now being installed in homes, apartment complexes, and businesses contain heat collecting and storing devices from which resources are drawn during non- and low-sunlight periods. Although there are many types of such systems, the most common circulate water or some other liquid through solar heating panels, or collectors, during the day and store the heated fluid in tanks. When required, this fluid is pumped through radiators or radiant coils to provide area heating. To maintain comfort and make the best use of the available energy, the user must continuously monitor the temperature of every area to be heated, as well as the temperatures of the collectors and the storage tanks. Valves must then be opened or closed and pumps turned on or off to maintain the desired relationship among the system components. A computerized energy management system, or solar controller, can perform all of the monitor and control functions with optimum efficiency.

A microprocessor-based solar controller designed by Fairchild for Rho Sigma, Inc., a major manufacturer of solar controls, is specifically intended to accept the low voltages produced by thermistor temperature sensors, process and display the data, and provide outputs for relay and switch opening and closing. The solar controller contains a single-board F8™ microprocessor, two input and two output cards, an A/D converter control card, a display control card, two 4K EPROM program storage cards, and a 1K RAM and memory address card (figure 1).

Also included in the unit are an A/D converter, a 5-digit LED display, and a 16-key keyboard. The display automatically sequences through all input channels, displaying the number and temperature of each channel for one second before cycling to the next. The keyboard can be used to halt this sequencing and either make the display continuously monitor only one channel or convert it to a clock-only display that shows time of day.

Controller Operation

The microprocessor is programmed to solve a set of logic/arithmetic equations. These equations are contained in the EPROM program storage, with the associated constants being held in the 1K RAM. The keyboard can be used to change a number of the equation constants, permitting system changes to be made without hardware modification.

In normal operation, the A/D converter receives analog temperature information from as many as 16 thermistors and presents the converted data to the microprocessor. Digital data, such as that produced by switch closures and teletype signals, can be presented directly to the microprocessor through the 16 digital inputs of the input cards. These data are used to solve the system functional equations and produce two types of microprocessor outputs.

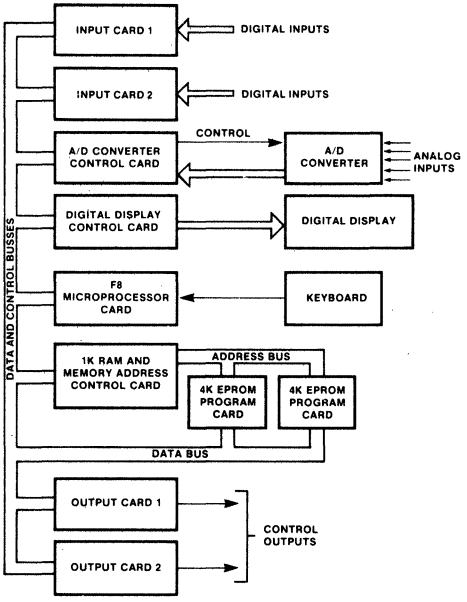
In the channel-monitor modes, temperature information is output to the display in degrees Fahrenheit or Celsius, depending upon resident program. In the time-display mode, a timekeeping routine program assumes control of the display circuitry and the temperature information is not provided.

The other microprocessor output consists of control signals that are suitable for opening and closing relays and activating solid state switches. These signals perform such functions as turning on pumps and opening valves to let water run into the storage tank or circulate through radiators.

Since program storage is in ROM, power failure does not cause catastrophic loss of memory. When power is restored, a resetting sequence begins, with the controller ensuring that all valves and controls are turned off so that stored energy is not lost. The controller then cycles through all of the inputs, decides what the system operating conditions should be, and generates the necessary output signals. This analysis takes approximately five seconds. To indicate to the user that power has been off, the display flashes until manually reset.

Originally designed for use in solar heating applications, the intelligent microprocessor-based controller is applicable to any system in which the ability to deal with multiple sensor inputs and generate control outputs is required.

Fig. 1 Solar Controller Functional Block Diagram



FAIRCHILD
A Schlumberger Company

HIGH-SPEED DATA ENCRYPTION USING THE F9414

INTRODUCTION

Until recently, the requirement of secrecy in data communications has been largely limited to military and diplomatic activities. Proprietary business, financial, and personal information was seldom transported through hostile environments; when necessary, suitable safeguards were provided by locks and, occasionally, guards. With the advent of solid-state electronics, the extensive use of standard radio links, microwave, commercial, and satellite radio channels, nearly anyone may listen in on masses of sensitive information without being detected.

Distributed computer systems with interactive terminals have increased effective communication between authorized users, but have also increased the problems of protecting the great quantities of confidential data from access by outsiders. The solution is to make the data somehow incomprehensible to unauthorized access through encryption.

In 1977, the National Bureau of Standards adopted the Data Encryption Standard, an encryption algorithm developed by researchers at IBM. The algorithm uses a 56-bit key to map one 64-bit data word into another, and is well-suited to LSI implementation. Accordingly, over the next few years, several semiconductor manufacturers announced LSI devices that perform the data encryption and decryption steps.

SIGNIFICANT CONCEPTS AND FEATURES OF THE F9414

Fairchild's F9414, a very fast encryption device, is suitable for extremely high bit-rate data communication and data storage applications. As this 4-chip device is aimed primarily at the performance-driven market, it is implemented with bipolar technology, blending low-power Isoplanar Integrated Injection Logic ($I^3L^{®}$) and high-speed TTL, made possible through Fairchild's proprietary isoplanar process. The speed-critical parts of the F9414, including the ROM lookup tables required by the DES algorithm, are TTL for minimum delay, while the control functions are I^3L for low power consumption. The TTL outputs ensure high speed and good drive capability. Added advantages of bipolar technology are high radiation resistance and full commercial and military operating temperature range.

From a manufacturing standpoint, a single-chip approach for the DES results in a chip size undesirably large for economical production of a high-volume device. However, the DES algorithm can be partitioned into a bit-slice implementation, yielding devices that are identical, with the exception of the internal ROM table. This approach results in a relatively small die size.

Data throughput of 13.3 MHz (75 ns) per bit can be achieved with a typical power consumption of 500 mW for V_{CC} and 150 mW at the injector pin, for a total of 650 mW per chip.

The F9414 chip set consists of four similar 40-pin devices. A typical connection is shown in figure 1. Figure 2 illustrates the major logic elements of one of the chips, including a pair of data registers, four 8-bit shift (key) registers, control logic, and two 64-bit word by 4-bit read-only memories (ROMs). The F9414 encryption set has passed the NBS functional validation test.

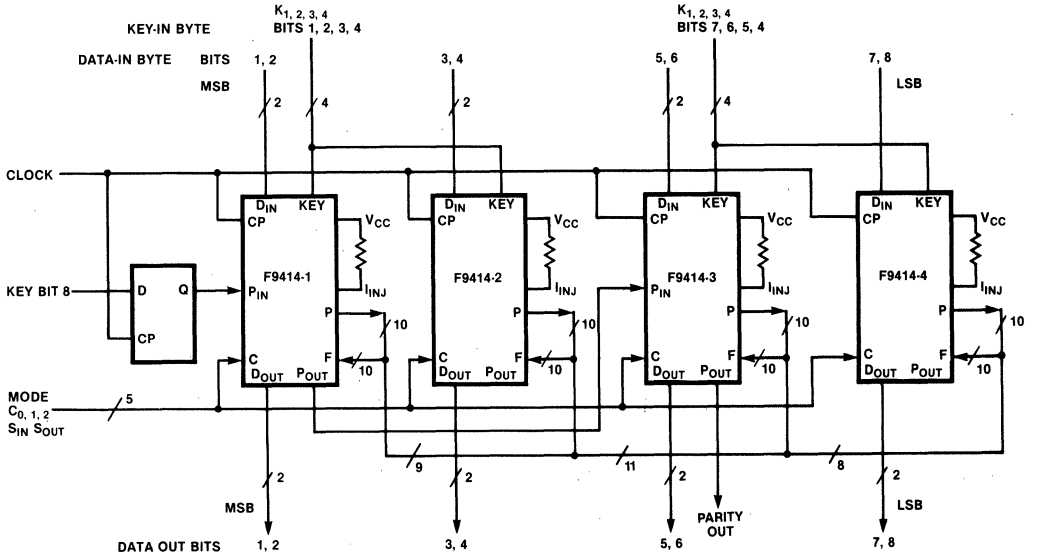
The set operates with a 56-bit key word to encipher or decipher a 64-bit data word that is stored in 8 bytes; 2 bits of each byte are distributed to each of the four chips. The key consists of 64 bits in 8 bytes; bit 8 of each byte is parity. Bits 1 through 4 go to both chip 1 and 2; bits 4 through 7 go to chips 3 and 4. The four chips together store the 64-bit plaintext or ciphertext word.

The chips have separate data inputs and outputs so the block of data to be processed can be input as the previous block is being output. This overlap permits the processing of a 64-bit block in 24 clock pulses, which, at a 5 MHz maximum clock frequency, makes the F9414 substantially the fastest of the present commercially available DES chip implementations. Data throughput of 13.3 MHz (75 ns) per bit or 200 kHz (4.8 μs) per 64-bit word can be achieved.

The key register is capable of hold, left shift (encipher), or right shift (decipher) operations, by one or two positions, as required by each of the 16 rounds of the algorithm. Each device also includes logic for the control of these registers during load and cipher operations. The 64-bit word by 4-bit ROMs in each device implement the S-boxes of the algorithm.

The major differences among the four devices are the masking of the ROM codes and the key bits that are selected as ROM addresses, according to the E-bit selection table of the algorithm. As shown in figure 1, a set of eight output signals ($P_{1\text{-}8}$) and input signals ($F_{1\text{-}8}$) is interconnected between chips to implement the permutation function, P, of the algorithm. An additional set of outputs, P_x and P_y, and inputs, F_x and F_y, is used to interconnect the chips as required by columns 1 and 6 of the E-bit selection table.

Figure 1 F9414 4-Chip Encryption Set



IMPLEMENTATION OF THE DES ALGORITHM

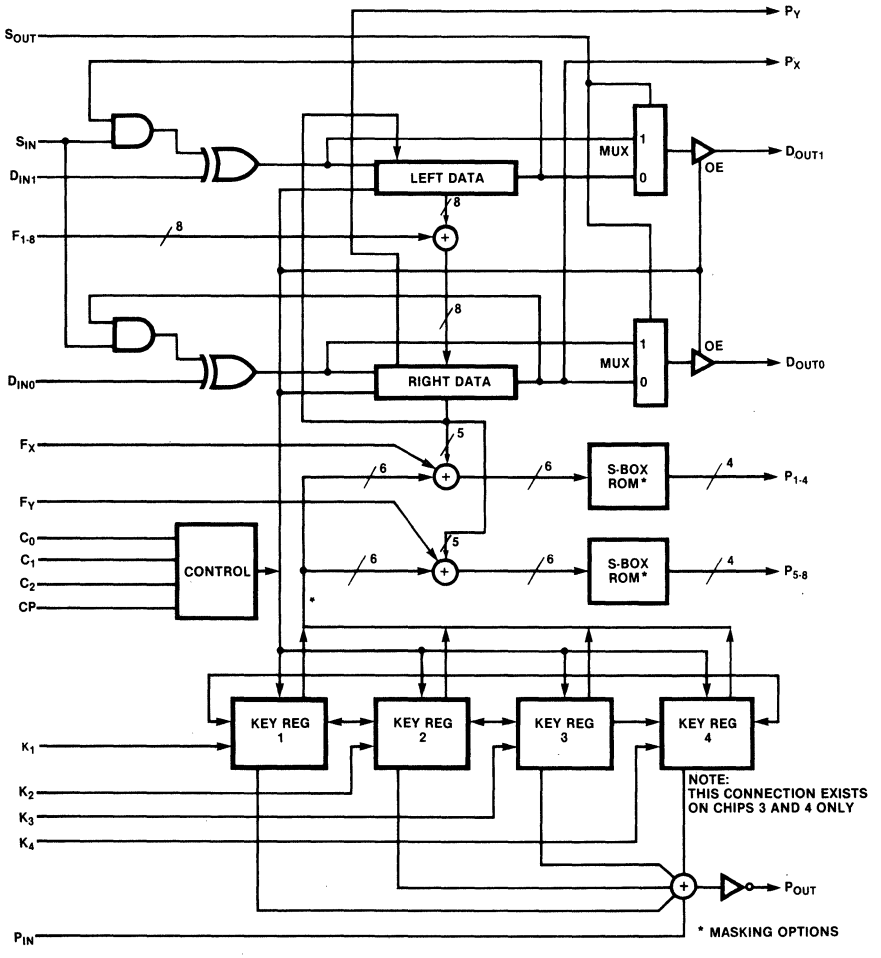
Initial permutation is accomplished in the F9414 chip set by the manner in which the data is loaded. The D_{IN0} input of chip 1 loads bit 1 of each byte, D_{IN1} of chip 1 loads bit 2 of each byte, D_{IN0} of chip 2 loads bit 3 of each byte, etc. After eight clock cycles, the four registers receiving data bits 2, 4, 6, and 8 of each input byte comprise the L_0 block of 32 bits in permuted order within the four devices (see figure 3). The four registers receiving bits 1, 3, 5, and 7 of each byte hold the R_0 block. Therefore, each chip slice contains one byte each of the L_0 and R_0 blocks.

Further shifting of the bits and extracting outputs from the right end of each byte implements the inverse permutation 1P-1. Each column of the inverse permutation may be found in a register byte and the first 8 bits (40, 8, 48, etc.) required by row 1 of the inverse permutation table are at the output ends of the shift registers.

The 28 key bits in the top half, C_0, of the key permutation function are duplicated in the key registers of F9414-1 and F9414-2, while key bits in the bottom half, D_0, occupy the registers of both the F9414-3 and F9414-4 (see figure 4). In each device, key register 4 holds the last 4 bits of both halves of the key permutation function. Each of the 16 iterations involves a left rotation (encipher) or right rotation (decipher) of the key registers.

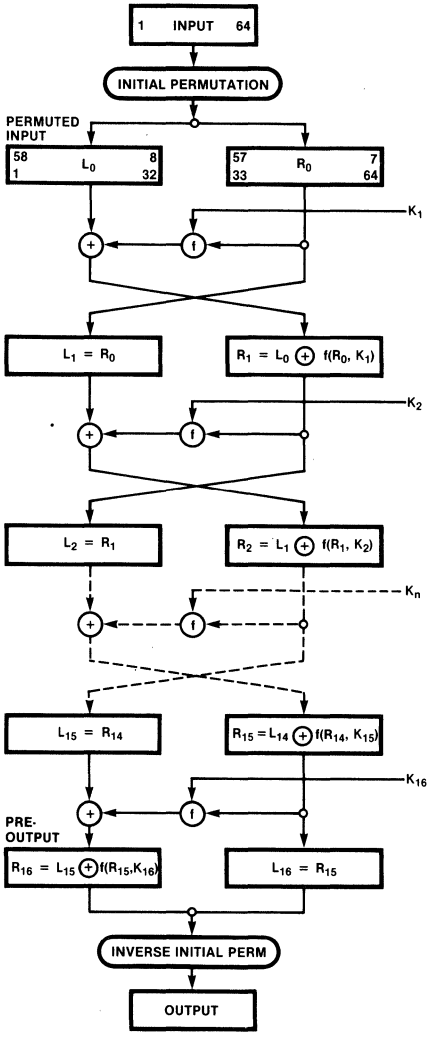
During the key shift schedule, chips 1 and 2 bypass the right half of key register 4, and chips 3 and 4 bypass the left. This results in the key alignment returning to its original position after a total of 28 shifts from the 16 alterations.

Figure 2 F9414 Block Diagram



10

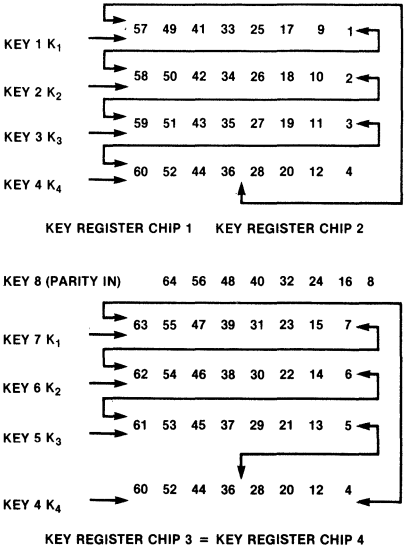
Figure 3 DES Algorithm



An internal 1-bit right realignment is required by a change from encipher to decipher after the key has been entered. This, and the reverse (left realignment for decipher to encipher), are performed by the F9414 control logic, which must be stable prior to the loading of the last data byte. When clocked at the same time as a load-key code, the data registers all fill with logic ones.

The results of the exclusive-OR of the key bits and data words derived from R_0 in the calculation of f(R, K) are taken, 6 bits at a time, to address a set of eight 64 × 4 S ROMs (i.e., S boxes). Two S ROMs per chip, each with four output bits, provide the 32 bits that are then permuted per primitive function P, by chip-to-chip interconnection. The effective result of the interconnect is exclusive-ORed with the L_0 block (figure 3) and the entire algorithm is repeated 16 times.

Figure 4 Key Permutations



The F9414 is structurally designed for high throughput. Since no I/O ports are used for both entering data and reading results, a potential bottleneck is avoided. The 64-bit data word is entered into the F9414 data registers one byte at a time at the D_0, D_1 inputs. The MSB of data goes to D_0 of the F9414-1. The result is output one byte at a time on the Q_0, Q_1 pins, MSB output first. Similarly, the keyword is entered one byte at a time at its own dedicated inputs (K_1-K_4). Table 1 shows the distribution of the keyword to the four F9414 devices.

Table 1 Keyword Distribution

Keyword	F9414-1 Key Reg.	F9414-2 Key Reg.	F9414-3 Key Reg.	F9414-4 Key Reg.
8 MSB	1	1		
7	2	2		
6	3	3		
5	4	4	4	4
4			3	3
3			2	2
2 LSB			1	1
1 Parity (Option)	P_{IN}			

The keyword is 56 bits long; if desired, an optional parity bit can be included with each byte of key, making the keyword 64 bits long. Parity does not in any way affect the encryption or decryption, and is taken across the keyword register, not across the K_1-K_4 inputs. Parity across one byte of keyword is taken by passing the parity bit of the keyword through a delay flip-flop to P_{IN} of the F9414-1 or F9414-2, and through P_{OUT} of the F9414-1 or F9414-2 into P_{IN} of the F9414-3 or F9414-4. The final parity sum is available on P_{OUT} of the F9414-3 or F9414-4.

The functions of the F9414 (load key, load data, encryption/decryption, and wait) are controlled by the C_0-C_2 inputs. Data and key are clocked in and/or out on low-to-high clock transitions. Loading a key sets the data registers to all high.

The F9414 enables simultaneous input and output of data; i.e., the results of a DES cipher operation can be clocked out on the same low-to-high transition that loads the next word to be processed. Thus, a complete input and output cycle (LOAD/READ DATA) takes just eight clocks. Since the algorithm requires 16 clocks, an entire DES iteration can be accomplished in 24 clocks. At a typical clock frequency of 6 MHz, this translates into a 16 MHz bit rate, a very fast LSI implementation of the DES. This high throughput ensures that the F9414 set is capable of keeping pace with practically every application, and this speed is available over the full military temperature range.

Several F9414 sets can be paralleled to produce higher bit rates. This implementation is facilitated by the F9414 3-state outputs, which automatically go into the high-impedance state when the device is executing the DES algorithm, and become active upon a READ DATA command following the completion of the algorithm. A complete iteration, including loading, executing the algorithm, and reading the data, takes 24 clocks. Since an F9414 set enables the outputs for only eight of these clocks, operating three sets, eight clocks out of phase with each other, ensures that there is no conflict at the outputs.

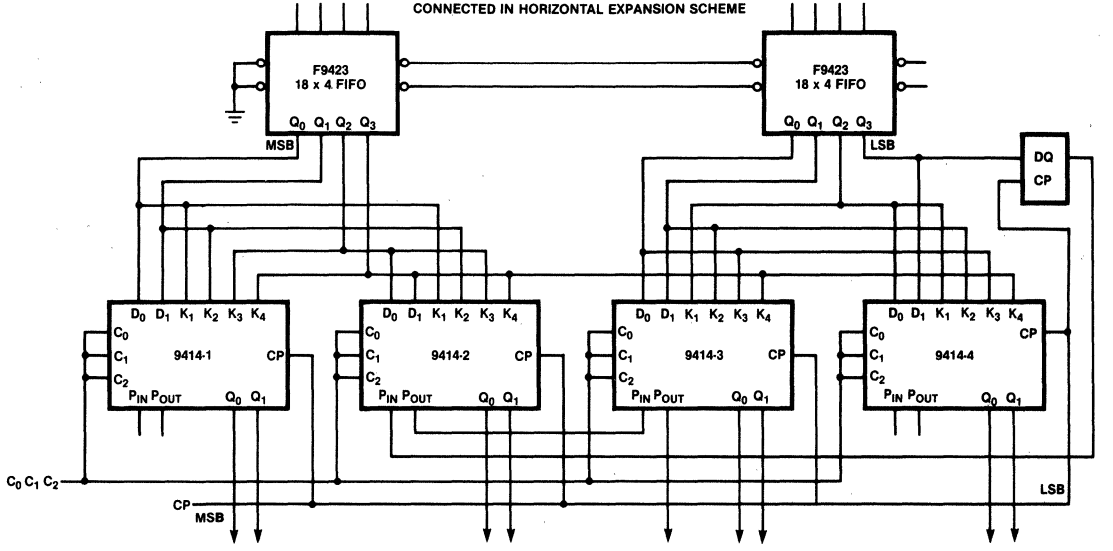
The independent byte-wide inputs and outputs of the F9414 set provide for simple integration into a bus-oriented system. The F9414 controls are similarly designed for simplicity and ease of use. The three C-lines are entirely responsible for the internal state of the device. To perform any one of the F9414 functions, it is only necessary to choose the appropriate code, ensuring that the setup time to the next clock edge has been met.

As an example, an asynchronous source, such as a modem, generates bytes to be supplied to the F9414 for decryption. Since the DES operates on 64 bits, the F9414 must wait until it has received 8 bytes before beginning a decryption. Buffering must be provided in case a byte should arrive while the F9414 is performing a decryption. One method is to use the F9414 set with two F9423 64 × 4 First-In First-Out (FIFO) buffer memories (see figure 5). The FIFOs provide buffering for incoming bytes. A low state on the F9423 Output Register Empty (ORE) signal indicates that the FIFO is empty. This signal can be used to put the F9414 into the wait state.

CIPHER FEEDBACK AND CIPHER BLOCK CHAINING APPLICATIONS

The NBS DES provides for data encryption on a one-to-one basis. Each 64-bit vector, for a given key, produces a unique output vector regardless of previous inputs. Two more sophisticated and secure approaches utilizing the DES are called cipher feedback (CFB) and cipher block chaining (CBC). Both variations form the output of the encryption operation as a function of previous inputs as well as the present input. Such techniques make it impossible to determine the keyword being used on the basis of a single data word and its enciphered counterpart, even by exhaustive search.

Figure 5 Parity, Data, and Keyword Connections for F9414 and F9423



Cipher Feedback

In cipher feedback (see figure 6), the present 64-bit data input is exclusive-ORed with the output of the encryption unit, and the result of this operation is transmitted and also fed back into the encryption unit to perpetuate the feedback. At the receiver, the received 64-bit vector is first exclusive-ORed and then deciphered.

Figure 7 illustrates the cipher feedback transmitter operation. A 64-bit buffer is needed for storing the input word external to the F9414, and can be provided with two F9423 FIFO buffer memories. Both receiver and transmitter operate in the same mode and start with the same (arbitrary) initialization word in the buffer. If the initialization is not done, the first 64 bits of data at the receiver are erroneously deciphered.

To encrypt 1 byte of data, one iteration of the DES algorithm is performed on the contents of the buffer. Then the MSB output from the F9414 is exclusive-ORed with the data byte and the result is transmitted. Additionally, the result of the exclusive-ORing is shifted into the least significant position of the buffer, while all other bytes are shifted and the former MSB discarded. This causes all following encryptions to depend on the present transmission, providing greater security than when each encryption depends only on the present data byte.

At the receiver (see figure 8), the transmission is shifted into the least significant position of the buffer and one DES iteration is performed. Since the receiver has used the same data word as the transmitter, this generates the same exclusive-OR mask as was used at the transmitter. Therefore, exclusive-ORing the next received byte with the MSB of the F9414 output recovers the data byte.

The transmitter and receiver must be operating in synchronization in cipher feedback. If synchronization is lost or an erroneous bit received, 64 bits of data will be incorrectly deciphered.

Figure 6 Cipher Feedback Mode

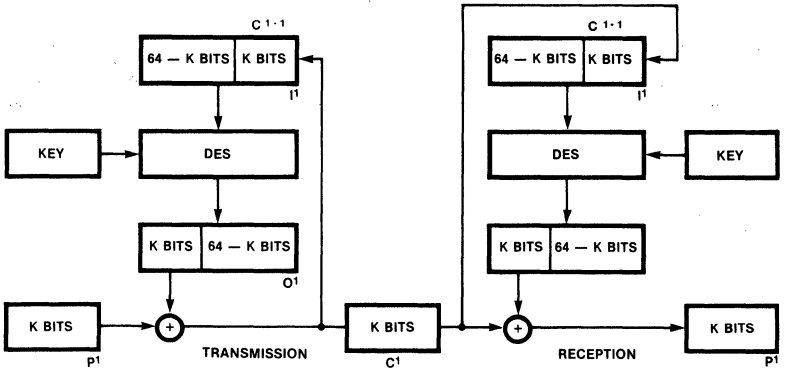
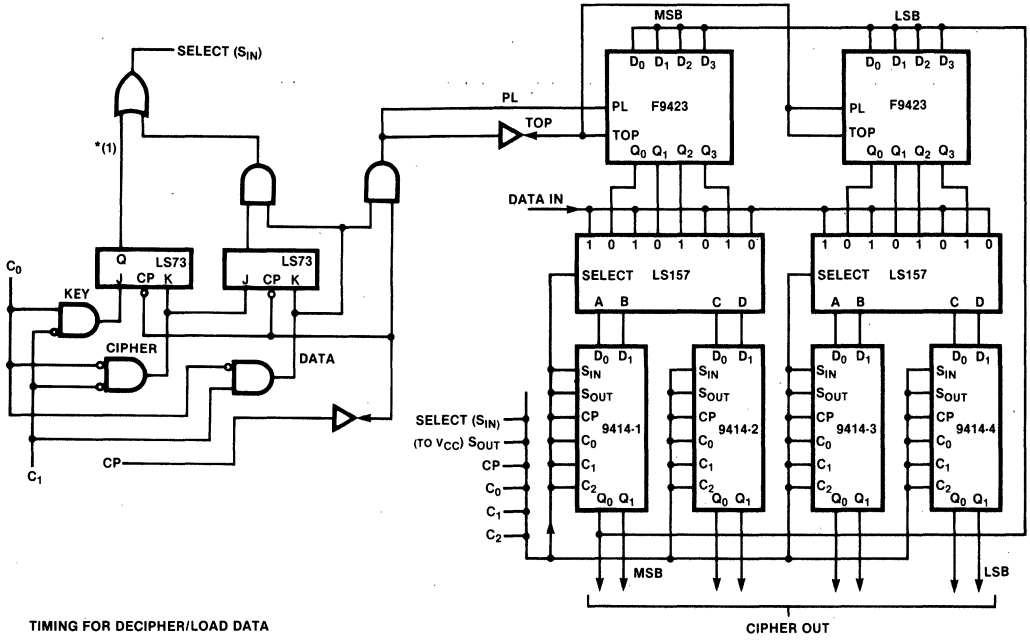
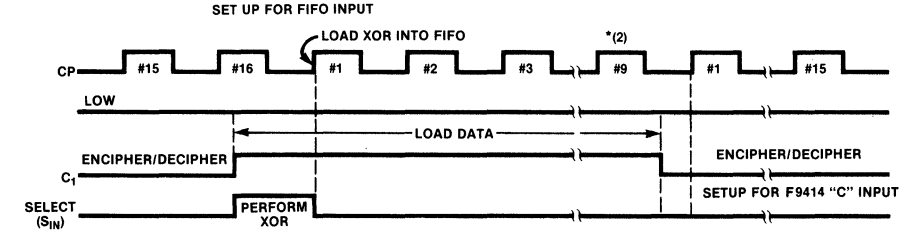


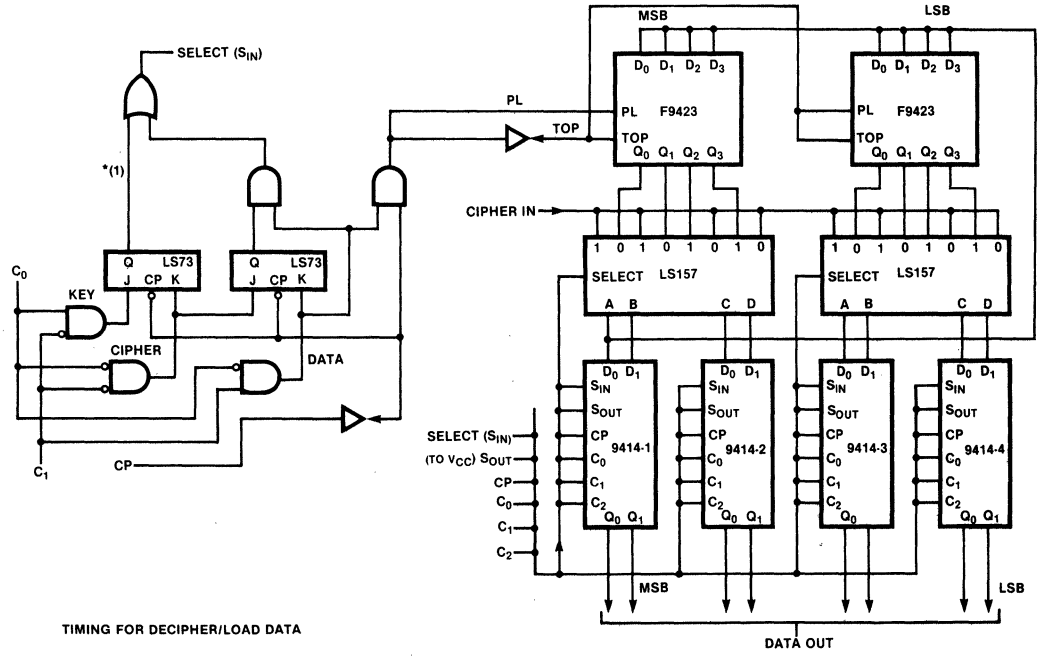
Figure 7 Cipher Feedback Transmitter



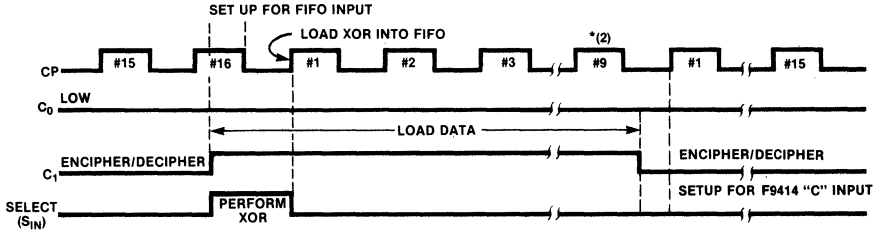


NOTES:
1. 8 byte initialization vector following LD KEY
2. 9 clock pulses required to complete load data operation

Figure 8 Cipher Feedback Receiver



TIMING FOR DECIPHER/LOAD DATA



NOTES:
1. 8 byte initialization vector following LD KEY
2. 9 clock pulses required to complete load data operation

10

Cipher Block Chaining

Cipher block chaining (see figure 9) is similar to cipher feedback in that successive transmissions are made dependent on previous transmissions, thereby increasing the level of security. The CBC transmitter takes the present 64-bit input vector and exclusive-ORs it with the output of the encryption unit, then performs an encryption and the result. The result of the encryption is transmitted and also exclusive-ORed with the next 64-bit vector, continuing the chaining process. The receiver runs synchronously with the transmitter and recovers the data by performing a decryption and then an exclusive-OR on the received 64 bits.

The receiver and transmitter must operate in different modes: encrypt and decrypt (see figures 10 and 11). No data buffering is necessary at the transmitter, but the receiver needs a 64-bit buffer to store the previous transmission. Both receiver and transmitter must start with the same initialization data or the first 64 bits of transmission will be incorrectly deciphered.

Internal exclusive-OR gates on the F9414 make implementation of the CBC transmitter especially simple. When S_{IN} is high (figure 10), the exclusive-OR of the D inputs and Q outputs is input to the F9414 register. Since the F9414 can input and output simultaneously, the input data and the F9414 output are exclusive-ORed while the result of the DES iteration is being clocked out at the Q outputs. Therefore, no additional packages are required.

SUMMARY

The F9414 4-chip data encryption set combines exceptionally high performance with general ease of use. Its speed across the full military temperature range makes it suitable for demanding applications.

Figure 9 Cipher Block Chaining Mode with Terminal Block Padding

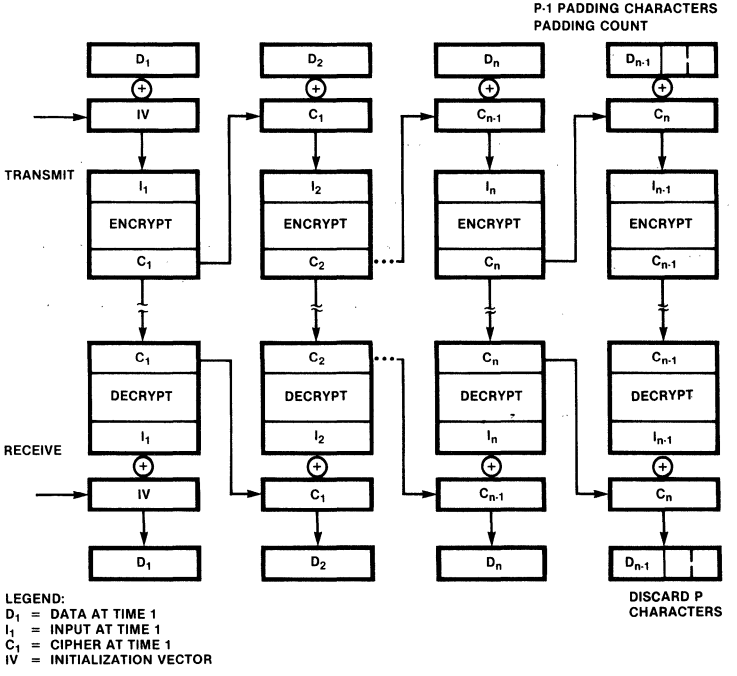
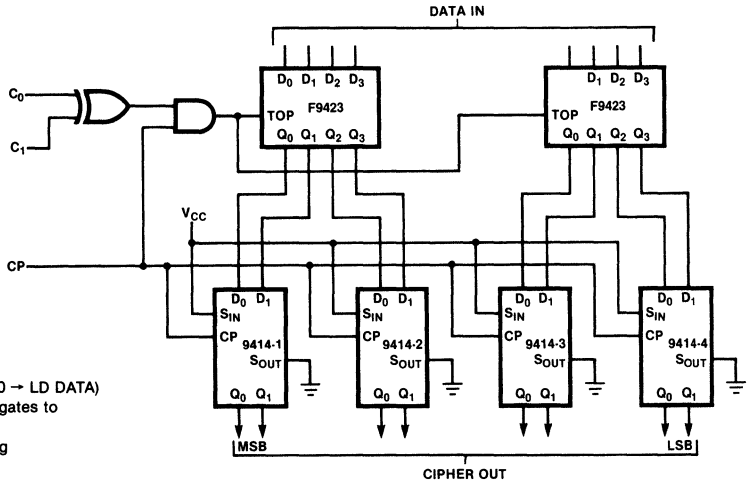
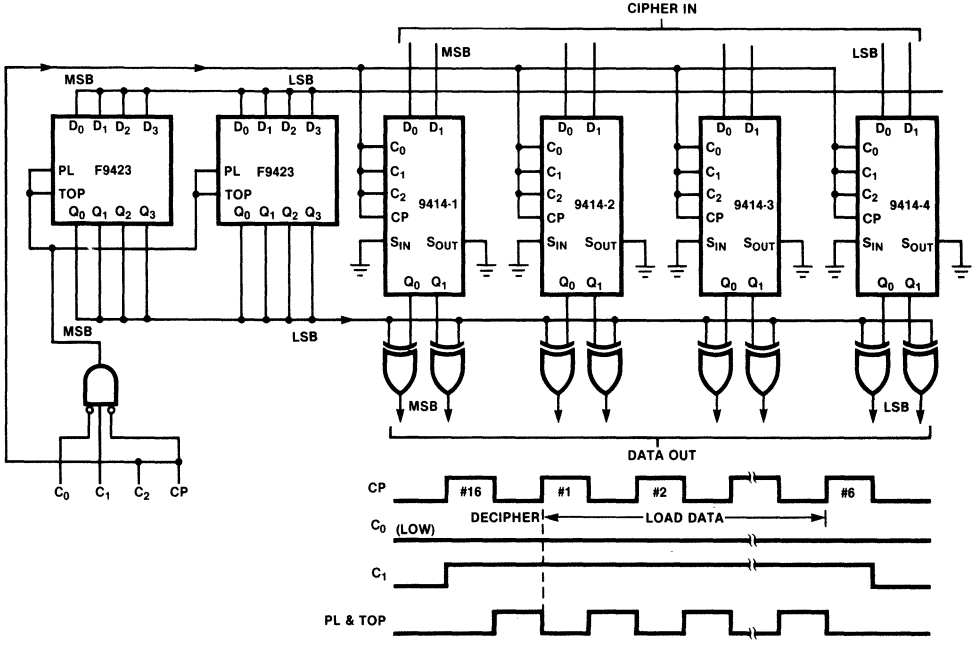


Figure 10 Cipher Block Chaining Transmitter



NOTES:
1. A high on S_{IN} (C_1 C_0 = 10 → LD DATA) enables the internal XOR gates to perform the chaining.
2. Hold S_{IN} low while loading initialization vector.

Figure 11 Cipher Block Chaining Receiver

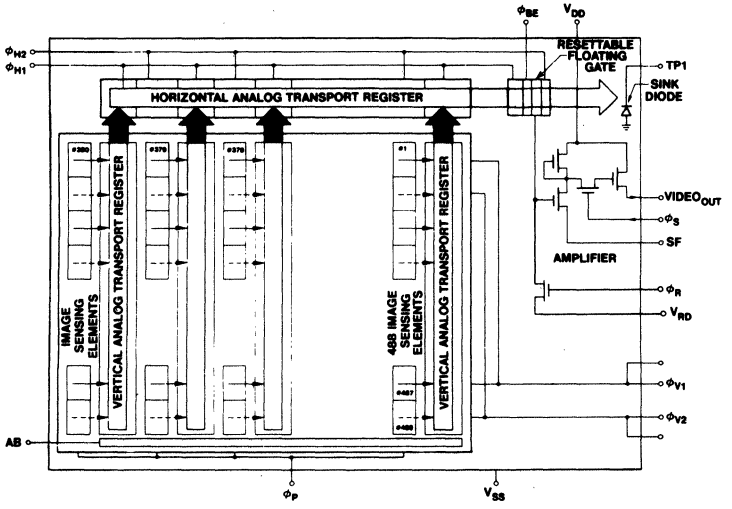


10

FAIRCHILD
A Schlumberger Company

Configuring a Binary Vision System with a CCD3000 Camera

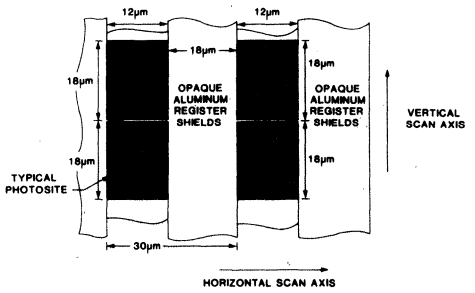
Figure 1 CCD Block Diagram



The CCD222 is organized as a matrix array of 488 horizontal lines by 380 vertical columns of charge-coupled photoelements. The dimensions of these 185,440 photoelements are 12 μm horizontally by 18 μm vertically. The photo elements are precisely positioned on 30 μm horizontal centers and 18 μm vertical centers. The CCD222 has an active optical area of 8.8 by 11.4 mm, with a diagonal of 14.4 mm.

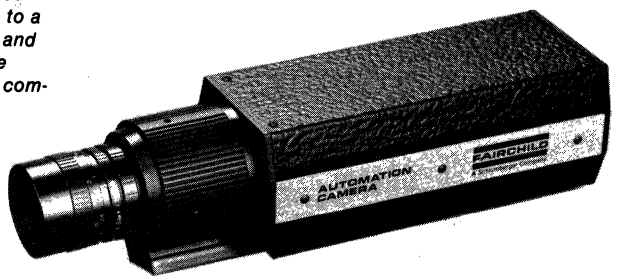
Figure 2 Structural Detail

The photosite dimensions of a CCD222 sensor are revealed in this structural detail. Note that photosite separator barriers are transparent, and photosites are optically contiguous along the vertical axis.



10

CCD Imaging offers several advantages over "past-generation" vacuum tube sensing, including size, weight, power, sensitivity and reliability. Equipping an automated process with vision requires interfacing a CCD camera to a computer providing instant, low cost, effective control and intelligence. As usual, there are certain obstacles to be overcome. Some are addressed here utilizing standard components and systems with minor modification.



Charge Coupled Devices

Charge-Coupled Devices (CCDs) are a family of silicon semiconductor components that operate on a principal called "charge-coupling." Charge-coupling is the collective transfer of mobile electric charge stored within a storage element, to a similar adjacent element by the external manipulation of applied voltages. CCDs are employed as memories, analog signal processors and imagers.

CCD image sensors sense a photon-generated charge in an array of depletion regions or potential wells formed by MOS-type capacitors. Then they serially transfer the charge from each element to an output charge detector amplifier. Sensors are classified as Linear Imaging Devices (LIDs) or Area Imaging Devices (AIDs).

Basically, an LID is composed of a row of image-sensing elements (photosites), two analog transport registers, and an output amplifier. Light energy falls on the photosites and generates charge packets proportional to the light intensity. These packets are then transferred in parallel to the analog transport registers and shifted by 2-phase clocks.

The charge packets are delivered to an on-chip amplifier and converted to a proportional voltage level. The output, a series of amplitude modulated pulses representing optical information, is called video.

AIDs are similar to LIDs except that the photosites are arranged in an x-y matrix (*Figure 1*) with opaque transport registers located between photosite columns (*Figure 2*). The charge packets are transferred to the output amplifier in two separate fields, line by line. This technique is called interline transfer.

The ability to generate, move about and detect discrete packets of electrons fulfills a number of image processing requirements. Characteristically, solid-state imaging exhibits features such as unity Gamma, zero lag and no geometric distortion. Conversely, the signal linearity produced by vidicons depends on uncertain analog sweep circuits which are a source of distortion and requires periodic calibration adjustments. Compared to other solid state photosensing arrays, CCDs possess a higher dynamic range (typically 1000:1) and charge transfer efficiency (.9999 typical CTE).

Fairchild CCD cameras are ideal in many scientific and industrial imaging applications that require a high degree of optical accuracy. Fairchild cameras have been successfully employed in these environments for several years. In part, this is possible because of the precise geometric pixel alignment and high quality linear video response inherent in each camera through careful circuit and enclosure design techniques. Small, light-weight, and rugged, CCD cameras are well suited for imaging in hostile environments normally too hazardous for traditional vidicon cameras.

Applications for CCD cameras (some are shown at the end of this article) include non-contact measurement, product inspection, color sorting, robot guidance, reconnaissance and surveillance.

As systems develop and improve, CCD Imaging applications broaden to include many diverse areas. The advantages of CCD cameras for celestial mapping and tracking are widely known and accepted in astronomy. Video data from scanned x-ray charts are commonly used for computer analysis in medicine. CCD cameras are used as navigational aids for pilots and provide visual information to both ground and airborne flight recording systems. Geological mapping of the earth's resources via satellite is an ongoing space application.

Recent software development in the area of artificial intelligence has led to "smart" vision systems capable of learned image and pattern recognition. Incorporated as the eyes for robots or automated processes, these systems are playing an ever-increasing role in mechanical assembly, quality control, welding and batch sorting tasks.

Most vision system implementations require at least one field of video data be stored in digital memory in real time for subsequent computer analysis. When imaging at standard TV video rates, fast memory and control circuitry are needed. Additionally, processing algorithms need to acquire stored video data as contiguous sequential video data lines, a difficult job to perform on interlaced video using hardware-based memory storage techniques.

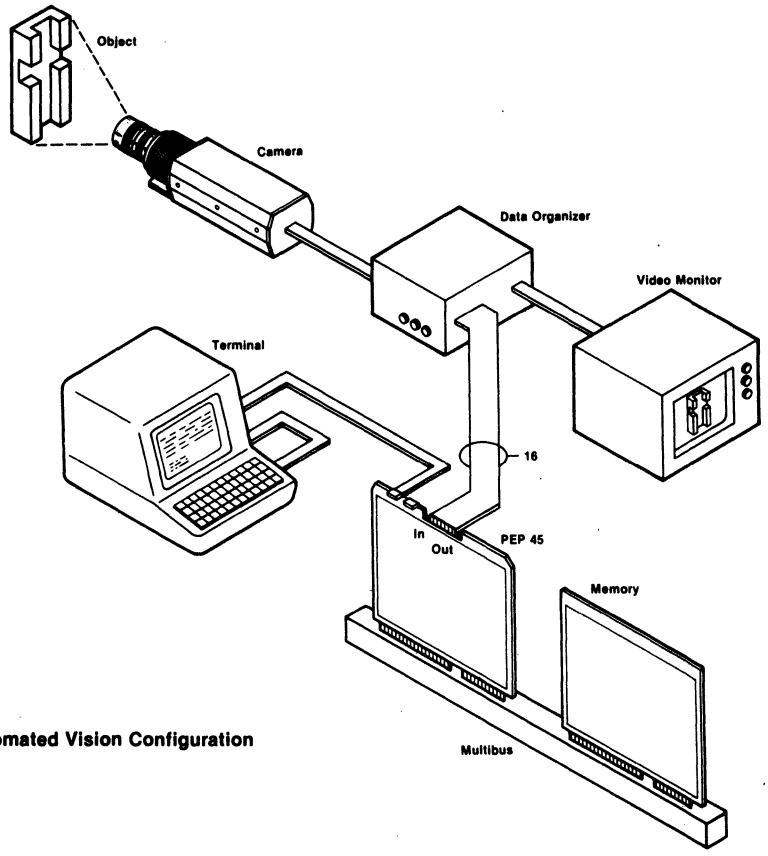


Figure 3 Typical Automated Vision Configuration

Intelligent Automation

Fairchild's CCD Imaging Group has demonstrated a Multibus* binary frame grabber that features the CCD3000 Video Communication Camera for image sensing, a PEP-45 Single-Board Computer as the system controller, and a 32 K byte static RAM multibus memory board for frame storage. A hardware interface, or "data organizer", connects the camera to the PEP-45 parallel input/output port. This system (*Figure 3*) can perform intelligent image and pattern recognition of high-resolution CCD images. As described, the system performs interlaced binary video frame grabbing with non-interlaced bit-mapped storage.

Imaging

Fairchild's CCD3000 and CCD4000 cameras are rugged, self-contained units that take advantage of the geometric accuracy, wide dynamic range, and reliability of a buried-channel charge coupled device image sensor. The CCD3000 Video Communications Camera provides a 380 element-per-line interlaced two-fields-per-frame NTSC composite video signal. The CCD4000 Automation Camera provides video in a non-interlaced 256-by-256 element square pixel pitch format required for certain automatic inspection, recognition, and robot guidance systems. Camera design makes them ideally suited for intelligent image processing systems typically found in robotic and automated applications. Either camera can be installed as a relatively small single unit, or separated into a sense head connected to the camera control electronics via cable, *Figure 4*.

Control

Several computer interface schemes exist for video cameras. Differences vary depending on the particular imaging application. Binary video is a 1-bit (2-level) digital representation of analog video relative to some dc threshold voltage. Digitized video usually refers to a two or more bit (multilevel) digital interpretation with respect to a voltage range. In either case, analog-to-digital conversion is required for frame storage. Direct Memory Access (DMA) techniques are often employed to allow video data acquisition in real time. Typically, this means custom-designed hardware for most off-the-shelf computer boards. Commercially available frame grabbers work well for vidicon cameras, but are not always readily adaptable to solid state cameras due to differences in image resolution and pixel data rates.

*Trademark of Intel Corporation

A PEP-45 single-board computer (*Figure 5*) from Fairchild's Microprocessor Division is used in this system. The PEP-45 uses a 20 MHz F9445 16-bit microprocessor. This board offers high throughput, multibus capability, and requires only a single +5 V power supply.

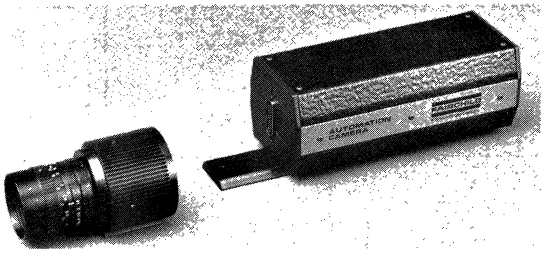


Figure 4 Sense Head Can Be Remote from Control Unit

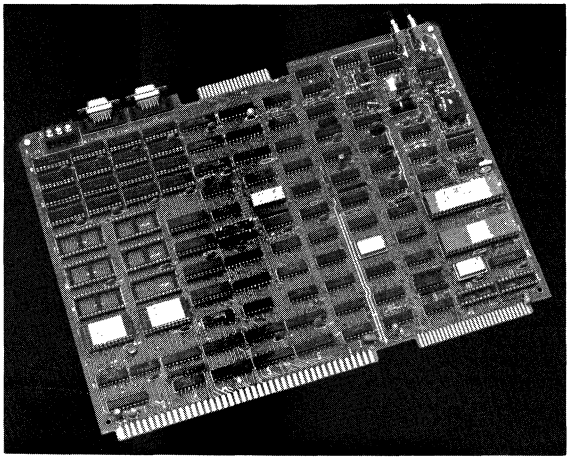


Figure 5 PEP-45 Single-Board Computer

System strategy is to first convert the analog signal to binary. Then, by using camera output timing signals, each video line is translated into a series of 24 data words, (*See Figure 6*). Each data word, consisting of 16 consecutive binary pixels, is fed to the memory mapped input port on the PEP-45 board. Upon reading the word serial-bit parallel video information under program control, the PEP-45 transfers the data over the multibus to memory.

RS 170 composite video from the CCD3000 camera consists of 525 horizontal lines in two fields, *Figure 7*. One field contains every other line of video data (the odd field) and the other field contains the intervening lines of video data (the even field). Generally, image processing routines performed on a stored picture frame are on a line-by-line basis. To facilitate such processing, the video image is stored non-interlaced. Thus, without additional processing, the image can be reconstructed by sequencing through memory.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
X	X	X	X	X	X	X	X
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
X	X	X	X	X	X	X	X
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24

Figure 6 Non-Interlaced Video Map

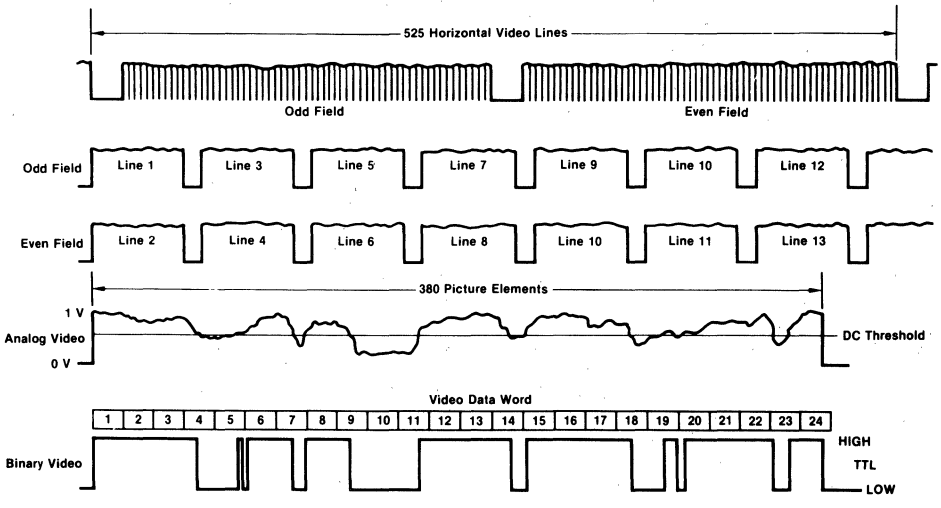


Figure 7 Video Timing

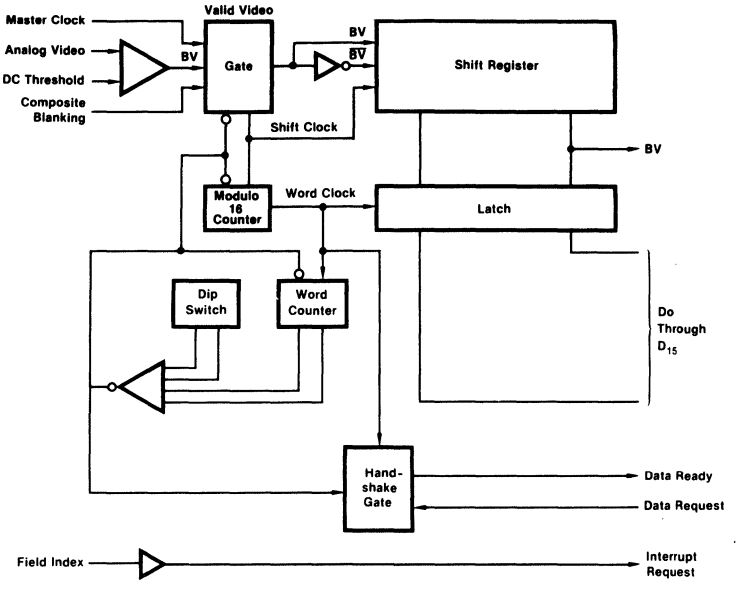


Figure 8 Data Organizer Block Diagram

Data Organizer Operation

The Data Organizer is used as an interface between the camera and the computer, *Figure 8*. Composite video from the camera is fed to an operational amplifier in the organizer and compared to a dc threshold voltage from a potentiometer setting to produce Binary Video (BV). The camera's 14.32 MHz master clock is divided in half to derive the video pixel data rate. Composite Blanking, a TTL signal used to indicate valid video, gates the pixel data clock, allowing it to function as a shift clock. With the occurrence of each shift clock pulse, binary pixel data is loaded into a serial-in/parallel-out shift register. Pixel data loaded on previous clock pulses is shifted over one bit, and a modulo-16 counter is incremented. A bit-line output from the register is connected to an oscilloscope or to a 75 Ω terminated video monitor. This signal provides real-time observation of Binary Video, allowing optimum setting of the comparator threshold.

The Word Clock output of the modulo-16 counter latches the shift register outputs, forming the bit-mapped data word every 2.236 μs during valid line times. Data Ready is activated by the Word Clock signal when the Data Request signal is present. Data Request is removed after the PEP-45 has successfully read the data word, returning Data Ready to the inactive state, completing a 2-wire handshake cycle.

Since the CCD3000 line resolution (380 pixels) is not an even multiple of 16, the last data word in each line must be shifted four places in order to maintain data consistency. The word counter output is compared to a DIP switch setting indicating the number of words in each video line. The comparator output resets all counters and logic gates in preparation for the next video line.

A buffered Field Index input indicates the start of a new frame (two fields). A reverse video switch connected to the shift register Select input sets the background/object polarity to facilitate image processing in either front-lighted or back-lighted situations.

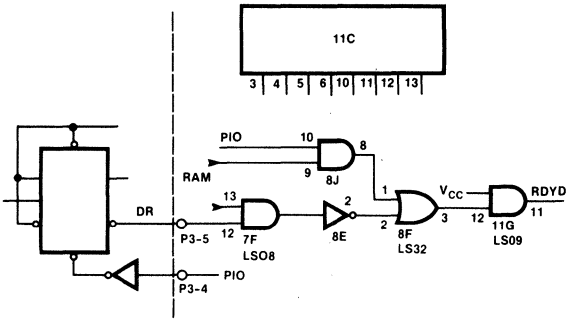


Figure 9 Speed Control Modification

The PEP-45 Board

Because the PEP-45 must acquire and store a video data word in 2.2 ns, instruction times for the image acquisition routine are critical. A minor board modification (*Figure 9*) was made to insure the fastest possible program execution. The 20 MHz F9445 microprocessor can execute an LDA (Load Accumulator) and a PSHA (Push Accumulator) instruction in 1.4 μs, leaving 800 ns to cover PEP-45 board and multibus overhead. Replacing the PEP-45 boards 16K words of 2114 NMOS RAM with fast 2148 RAMs for running the camera service routine eliminates the need for extended memory cycles.

Polling the status port to determine if the camera data is ready would require an additional instruction. By allowing the Data Ready signal from the data organizer to control the processor's RDYD line when the camera is accessed, video data acquisition time is minimized. Using available gates on the PEP-45 board, the RDYD delay circuit is reconfigured as shown in *Figure 9*. Signals ST_0 and CM_1 are removed from the P3 connector and replaced by RDYD and PIO respectively. See *Table 1*.

A 32 K byte, 220 ns cycle time static memory card is placed on the IEEE 796 bus and the remaining hardware configured. Once the program is loaded, the system is ready to operate.

PEP-45 Board Operation

When the processor is interrupted, the image acquisition routine is called. The Stack Pointer, line, and field count registers are initialized. A total of 24 sets of LDA and PSHA instructions are then executed, retaining the first line of binary video in the odd field. During the horizontal blanking period (the time in between valid video lines), the line count is updated. The Stack Pointer is adjusted, leaving enough

PEP-45 P3 Connector		CCD Data Organizer J5 Connector
Signal	**Pin**	**Signal**
IN0	13	D15
IN1	14	D14
IN2	11	D13
IN3	12	D12
IN4	9	D11
IN5	10	D10
IN6	7	D9
IN7	8	D8
IN8	21	D7
IN9	22	D6
IN10	19	D5
IN11	20	D4
IN12	17	D3
IN13	18	D2
IN14	15	D1
IN15	16	D0
RDYD	5	DATA READY
XINT	1	INTERRUPT REQUEST
PIO	4	DATA REQUEST

Table 1. Data Organizer/PEP-45 Connections

memory between the previous and present stack addresses for storage of the first video line from the alternate (even) field. Then, the program returns to acquire the next line of data in the field being acquired and stored (odd field). After the odd field has been collected, the cycle is repeated for the even field with the data acquired interlaced with data already stored from the odd field. Once the total picture frame is stored in memory, the PEP-45 (or any other processor sharing the multibus) can access the image data, *non-interlaced*, for further processing.

Summary

The CCD3000/PEP-45 combination demonstrates key capabilities for two diverse Fairchild products - a high-resolution CCD image containing over 180,000 picture elements being captured at NTSC video rates by a fast 16-bit microprocessor board. Superficially, the F9445 microprocessor performs a mundane role usually assigned to hardware logic. However, adept use of the board's powerful instruction set can convert the system function from that of a frame-grabbing camera interface to an extremely capable image processor. Additionally, interactive console operation, FS-1 linkage, PEP-BUG and PEP-BASIC readily support more sophisticated application development. Multibus-configured distributed processing schemes with other IEEE-796 masters are feasible.

FAIRCHILD
A Schlumberger Company

Section 11
Microprocessor Resource and Training Centers

Microprocessor Resource Center

The Microprocessor Resource Center (MRC) organization was created to serve as yet another, technically oriented, Fairchild customer link to a world-wide support structure that is concerned with all phases of the customer's requirements.

Every MRC is available to assist the customer in microprocessor hardware and software development and application engineering, product definition, and long-term product strategies. Backed by Fairchild's expertise and extensive resources, the MRC is a tool for solving current problems and planning for future needs.

At each Center is a microprocessor expert who is equally familiar with standard devices and those Fairchild state-of-the-art products that are on the leading edge of technology. Because they understand the microprocessor market and development trends, these experts provide technical support and planning assistance that can benefit the customer through timely and cost-effective system design and implementation.

As an added convenience, all Microprocessor Division development systems can be demonstrated at the MRCs. This affords the customer an opportunity to assess the performance and applicability of products in an operational-type environment.

Training on Fairchild's microprocessor products is available at either the customer or MRC location. This training, which is coordinated by the MRC manager, is performed in conjunction with the Microprocessor Division Education Center.

MRC Addresses

Jim Gunn
1702 Collins Blvd., Suite 101
Richardson, TX 75081
(214) 234-3391
TWX: DALL

Ralph Hayhurst
500 Park Blvd., Suite 575
Itasca, IL 60143
(312) 773-3300
TWX: ROLL

Karl Kulp
4570 W. 77th Street, Suite 356
Edina, MI 55435
(612) 835-3322
TWX: MINN

Rich Morse
5 Speen Street
Framingham, MA 01701
(219) 483-6453
TWX: FHAM

Gene Price
1570 Brookhollow, Suite 206
Santa Ana, CA 92705
(714) 557-7350
TWX: SANA

Microprocessor Education Center

Education plays a key role in the technical support of a microprocessor user. It is essential to a full understanding of the complexities, capabilities, and applications of modern processor products, and is therefore treated as an important component of the Fairchild Microprocessor Division customer support structure.

The most recent advances in microprocessor technology are included in the Fairchild Microprocessor Eduction Center courses, which feature a maximum amount of hands-on experience. Indeed, the major thrust of the training courses is to focus on the general techniques of microprocessor usage. This emphasis, it is felt, best prepares the student to apply the available design and development tools to specific applications in an efficient manner.

The following Education Center course offerings are included.

- **F8 and F3870 Microprocessor Systems**
  Intended to introduce the student to the Fairchild F8 and F3870 microprocessor systems, this course provides basic knowledge of F8 and F3870 hardware, software, applications, and development aids. Included are laboratory sessions in which the student applies that knowl-

edge in practical situations. Among the areas covered are device features and architecture, use of the various registers, machine and assembly language syntax, program writing and debugging, and hands-on use of the PEP 38 and Formulator systems.

- **F6800 Microprocessor Family**
  This course provides the student basic knowledge of Fairchild F6800 8-bit microprocessor family hardware, software, applications, and development aids. Included are laboratory sessions in which the student applies that knowledge in practical situations. Among the areas covered are device features and architecture, register organization and use, system configurations, program writing and debugging, and hands-on use of the various training and development aids.

- **F9445 Family Introduction**
  This course is an overview of the Fairchild F9445 16-bit microprocessor and its supporting circuits. Consisting of both lecture and laboratory sessions, with emphasis on hands-on experience, the course covers such areas as F9445 CPU and system timing, software, device features and architecture, and use of the FS-I and EMUTRAC development aids.

- **F16000 Family Introduction**
  Introducing the student to the Fairchild F16000 16-bit microprocessor family, this course is an overview that consists of both lecture and laboratory sessions. Emphasizing hands-on experience in the laboratory, the course covers such areas as device features and architecture, CPU and system timing, principles of memory management and virtual memory, floating point arithmetic, and familiarization with design aids.

- **FS-I Development System**
  This course introduces the student to the Fairchild System-I (FS-I) development system, emphasizing hands-on experience. Included is coverage of operating system usage, utility software usage, high-level languages and their associated compilers and interpreters, and the EMUTRAC emulation and tracking system.

- **Microprocessor Control and Interface**
  This course is intended to introduce the student to the principles and techniques of microprocessor control and interfacing. Opportunity is provided for hands-on experimentation with a mini-development system. Included in the course are a review of microprocessor fundamentals, transducer types and applicability, conversion techniques, and parallel and serial formats.

- **Pascal for Microprocessors**
  An introduction to the high-level Pascal language, this course teaches the student the skills required to produce software in Pascal for many practical applications, including real-time computing, scientific and engineering-type problem solving, and data processing.

FAIRCHILD
A Schlumberger Company

Sales Offices

United States and Canada

Alabama
Huntsville Office
500 Wynn Drive, Suite 511
Huntsville, Alabama 35805
Tel: 205-837-8960

Arizona
Phoenix Office
2255 West Northern Road, Suite B112
Phoenix, Arizona 85021
Tel: 602-864-1000 TWX: 910-951-1544

California
Los Angeles Office*
Crocker Bank Bldg.
15760 Ventura Blvd., Suite 1027
Encino, California 91436
Tel: 213-990-9800 TWX: 910-495-1776

San Diego Office*
4355 Ruffin Road, Suite 100
San Diego, California 92123
Tel: 714-560-1332 TWX: 910-335-1512

Santa Ana Office*
1570 Brookhollow Drive, Suite 206
Santa Ana, California 92705
Tel: 714-557-7350 TWX: 910-595-1109

Santa Clara Office*
3333 Bowers Avenue, Suite 299
Santa Clara, California 95051
Tel: 408-987-9530 TWX: 910-338-0241

Colorado
Denver Office
1200 E. Girard Ave., Suite 222, Bldg. B
Denver, Colorado 80231
Tel: 303-695-4927

Connecticut
Meriden Office
250 Pomeroy Avenue, Suite 202
Meriden, Ct. 06450
Tel: 203-634-8722

Florida
Ft. Lauderdale Office
5237 N.W. 33rd Ave. Suite 2D
Ft. Lauderdale, Florida 33309
Tel: 305-485-7970 TWX: 510-955-4098

Orlando Office*
Crane's Roost Office Park
399 Whooping Loop
Altamonte Springs, Florida 32701
Tel: 305-834-7000 TWX: 810-850-0152

Georgia
Atlanta Sales Office
3220 Pointe Pkwy, Suite 1200
Norcross, GA 30092
Tel: 404-441-2740

Illinois
Itasca Office
500 Park Blvd., Suite 575
Itasca, Illinois 60143
Tel: 312-773-3300

Indiana
Indianapolis Office
7202 N. Shadeland, Room 205
Castle Point
Indianapolis, Indiana 46250
Tel: 317-849-5412 TWX: 810-260-1793

Iowa
Cedar Rapids Office
373 Collins Road, NE, Suite 200
Cedar Rapids, Iowa 52402
Tel: 319-395-0090

Kansas
Kansas City Office
8600 West 110th Street, Suite 209
Overland Park, Kansas 66210
Tel: 913-649-3974

Maryland
Columbia Office
1000 Century Plaza, Suite 225
Columbia, Maryland 21044
Tel: 301-730-1510 TWX: 710-826-9654

Massachusetts
Framingham Office
5 Speen Street
Framingham, Massachusetts 01701
Tel: 617-872-4900 TWX: 710-380-0599

Michigan
Detroit Office*
21999 Farmington Road
Farmington Hills, Michigan 48024
Tel: 313-478-7400 TWX: 810-242-2973

Minnesota
Minneapolis Office*
4570 West 77th Street, Room 356
Minneapolis, Minnesota 55435
Tel: 612-835-3322 TWX: 910-576-2944

New Jersey
New Jersey Office
Vreeland Plaza
41 Vreeland Avenue
Totowa, New Jersey 07511
Tel: 201-256-9006

New Mexico
Albuquerque Office
North Building
2900 Louisiana N.E. South G2
Albuquerque, New Mexico 87110
Tel: 505-884-5601 TWX: 910-379-6435

New York
Fairport Office
815 Ayrault Road
Fairport, New York 14450
Tel: 716-223-7700

Hauppauge Office
300 Wheeler Road, Suite 201
Hauppauge, New York 11788
Tel: 516-348-0900

Poughkeepsie Office
19 Davis Avenue
Poughkeepsie, New York 12603
Tel: 914-473-5730 TWX: 510-248-0030

North Carolina
Raleigh Office
1100 Navaho Drive, Suite 112
Raleigh, North Carolina 27609
Tel: 919-876-9643

Ohio
Dayton Office
5045 North Main Street, Suite 105
Dayton, Ohio 45414
Tel: 513-278-8278 TWX: 810-459-1803

Cleveland Office
6133 Rockside Rd., Suite 407
Cleveland, Ohio 44131
Tel: 216-447-9700

Oklahoma
Tulsa Office
9810 East 42nd Street, Suite 127
Tulsa, Oklahoma 74145
Tel: 918-627-1591

Oregon
Portland Office
8196 S.W. Hall Blvd., Suite 328
Beaverton, Oregon 97005
Tel: 503-641-7871 TWX: 910-467-7842

Pennsylvania
Philadelphia Office*
2500 Office Center
2500 Maryland Road
Willow Grove, Pennsylvania 19090
Tel: 215-657-2711

Tennessee
Knoxville Office
Executive Square II
9051 Executive Park Drive, Suite 502
Knoxville, Tennessee 37923
Tel: 615-691-4011

Texas
Austin Office
8240 Mopac Expressway, Suite 270
Austin, Texas 78759
Tel: 512-346-3990

Dallas Office
1702 North Collins Street, Suite 101
Richardson, Texas 75081
Tel: 214-234-3391 TWX: 910-867-4757

Houston Office
9896 Bissonnet-2, Suite 470
Houston, Texas 77036
Tel: 713-771-3547 TWX: 910-881-8278

Canada
Toronto Regional Office
2375 Steeles Avenue West, Suite 203
Downsview, Ontario, M3J 3A8, Canada
Tel: 416-665-5903 TWX: 610-491-1283

FAIRCHILD
A Schlumberger Company

Sales Offices

International

Fairchild European Design Center
SHIRE HALL
Shinfield Park
Reading, England
Tel: 0734752175

Australia
Fairchild Austr. Pty. Ltd.
Suite 1, First Floor
366 White Horse Road
Nunawading, Victoria 3131
Tel: 61-3-8775444 Telex 36496

Austria
Fairchild Electronics GmSH
Meidlinger Hauplstr. 46
Tel: (0222) 858682/83 Telex 075096

Benelux
Fairchild Semiconductor
Ruysdaelbaan 35
5613 Dx Eindhoven
The Netherlands
Tel: 00-31-40-446909 Telex: 00-1451024

Brazil
Fairchild Semiconductores Ltda.
Caixa Postal 30407
Rua Alagoas, 663
01242 Sao Paulo, Brazil
Tel: 66-9092 Telex: 011-23831
Cable: FAIRLEC

France
Fairchild Camera & Instrument S.A.
121, Avenue d'Italie
75013 Paris, France
Tel: 331-584-55 66 Telex: 0042 200614 or 260937

Germany
Fairchild Camera and Instrument GmBH
Daimlerstrasse 15
8046 Garching Hochbruch
Munich, Germany
Tel: (089) 320031 Telex: 52 4831 fair d

Fairchild Camera and Instrument GmBH
Oeltzenstrasse 15
3000 Hannover
W. Germany
Tel: 0511 17844 Telex: 09 22922

Fairchild Camera and Instrument GmBH
Poststrasse 37
7251 Leonberg
W. Germany
Tel: 07152 41026 Telex: 07 245711

Hong Kong
Fairchild Semiconductor HK Ltd.
135 Hoi Bun Road
Kwun Tong
Kowloon, Hong Kong
Tel: 3-440233 and 3-890271
Telex: HKG-531

Italy
Fairchild Semiconducttori, S.P.A.
Via Flamenia Vecchia 653
00191 Roma, Italy
Tel: 06 327 4006 Telex: 63046 (FAIR ROM)

Fairchild Semiconducttori S.P.A.
Viale Corsica 7
20133 Milano, Italy
Tel: 296001-5 Telex: 843-330522

Japan
Fairchild Japan Corporation
Pola Bldg.
1-15-21, Shibuva
Shibuya-Ku, Tokyo, 150, Japan
Tel: 03 400 8351 Telex: 242173

Fairchild Japan Corporation
Yotsubashi Chuo Bldg.
1-4-26, Shinmachi
Nishi-Ku, Osaka 550, Japan
Tel: 06-541-6138/9

Korea
Fairchild Semikor Ltd.
K2 219-6 Gari Bong Dong
Young Dung Po-Ku
Seoul 150-06, Korea
Tel: 85-0067 Telex: FAIRKOR 22705

(mailing address)
Central P.O. Box 2806

Scandinavia
Fairchild Semiconductor AB
Svartengsgatan 6
S-11620 Stockholm
Sweden
Tel: 8-449255 Telex: 17759

Singapore
Fairchild Semiconductor Pty. Ltd.
No. 11, Lorong 3
Toa Payoh
Singapore 12
Tel: 531-066 Telex: FAIRSIN-RS 21376

Taiwan
Fairchild Semiconductor Ltd.
Hsietsu Bldg., Room 502
47 Chung Shan North Road
Sec. 3 Taipei, Taiwan
Tel: 573205 thru 573207

United Kingdom
Fairchild Camera and Instrument Ltd.
Semiconductor Division
230 High Street
Potters Bar
Hertfordshire EN6 5BU
England
Tel: 0707 51111 Telex: 262835

Fairchild Semiconductor Ltd.
17 Victoria Street
Craigshill
Livingston
West Lothian, Scotland-EH54 5BG
Tel: Livingston 0506 32891
Telex: 72629

FAIRCHILD

A Schlumberger Company

Printed in U.S.A./4027001 20M January 1983