UNISYS

CTOS®
BMULTI
Operations and Programming Guide

April 1991

Printed in U S America
3915 4133-000

Priced Item

UNISYS

CTOS®
BMULTI
Operations and Programming Guide

April 1991

Printed in U S America
3915 4133-000

Priced Item

Correspondence regarding this publication should be forwarded directly to Unisys Corporation, Round Valley Development Center, Route 22 & Werner Way, Lebanon, NJ 08833, U.S.A.

Page Status

Page	Issue
iii through xix	Original
xx	Blank
xxi	Original
xxii	Blank
1–1 through 1–6	Original
2–1 through 2–9	Original
2–10	Blank
3–1 through 3–14	Original
4–1 through 4–32	Original
5–1 through 5–4	Original
6–1 through 6–10	Original
7–1 through 7–6	Original
8–1 through 8–16	Original
9–1 through 9–17	Original
9–18	Blank
10–1 through 10–10	Original
11–1 through 11–19	Original
11–20	Blank
12–1 through 12–9	Original
12–10	Blank

Page	Issue
A–1 through A–9	Original
A–10	Blank
B–1 through B–2	Original
C–1	Original
C–2	Blank
D–1 through D–3	Original
D–4	Blank
E–1 through E–9	Original
E–10	Blank
F–1 through F–2	Original
G–1 through G–16	Original
Glossary–1 through 7	Original
Glossary–8	Blank
Index–1 through 7	Original
Index–8	Blank

Unisys uses an 11–digit document numbering system. The suffix of the document number (1234 5678–xyz) indicates the document level. The first digit of the suffix (x) designates a revision level when the revision incorporates only previous updates and no other editorial or technical changes; the second digit (y) designates an update level. For example, the first release of a document has a suffix of –000. A suffix of –130 designates the third update to revision 1. The third digit (z) is used to indicate an errata for a particular level and is not reflected in the page status summary.

About This Guide

The CTOS® BMULTI communications service allows CTOS workstations and shared resource processors to communicate with Unisys A-Series and V-Series mainframe systems, using a poll/select multipoint data communications protocol.

Purpose

This guide enables system administrators and programmers to install BMULTI, operate the BMULTI line and status monitors, and write programs that use the BMULTI data communications service.

Scope

This guide describes loading, configuring, installing, and operating BMULTI. It also provides information for writing programs that use the BMULTI programmatic interface.

Audience

The audience for this guide is the system administrator who wants to install and operate BMULTI. The system administrator can use sections of this guide to assist operators (non–programmers) in configuring and installing BMULTI. This guide is also for the use of programmers who want to write applications that use BMULTI services.

Prerequisites

System administrators should be familiar with CTOS systems. Programmers should be familiar with CTOS systems, as well as the Editor, Debugger, Linker, and the languages and compilers supported by CTOS.

How to Use This Guide

This guide provides an overview of the BMULTI product and information necessary to configure and install it. This guide explains the use of the BMULTI line and status monitors. This guide also describes the BMULTI programmatic interfaces, which are used for writing programs that use the BMULTI data communications service.

Organization

This Guide consists of 12 sections and 7 appendixes:

Section 1. Overview

This section. provides a general introduction to BMULTI and a description of its major features.

Section 2. System Requirements

This section lists the hardware and software requirements for using BMULTI, as well as other system constraints that may affect the use of BMULTI.

Section 3. Software Installation

This section describes the steps to load the BMULTI software from the distribution disks onto the hard disk of a CTOS workstation or shared resource processor, and lists the files copied and the commands created.

Section 4. Configuring and Installing BMULTI

This section describes the tasks of configuring BMULTI to operate on a CTOS system, using the BMULTI Configuration File Editor, and loading BMULTI into RAM as a system service.

Section 5. The Status Monitor

This section describes how to use the BMULTI status monitor utility.

Section 6. The Line Monitor

This section describes how to use the BMULTI line monitor utility.

Section 7. Basic BMULTI Concepts

This section contains recommendations for programming BMULTI and explains basic concepts.

Section 8. BMULTI State Machine

This section provides a conceptual tool for understanding how previously executed commands and events affect possible actions.

Section 9. Protocol Description

This section describes BMULTI protocol, the rules and conventions by which CTOS workstations communicate through BMULTI with Unisys A-Series and V-Series mainframe systems.

Section 10. High–Level Procedural Interface

This section lists the High–Level Interface (HLI) BMULTI procedures. Of the two BMULTI interface levels, HLI is the easiest to use.

Section 11. Low–Level Procedural Interface

This section lists the Low–Level Interface (LLI) BMULTI procedures. Although more difficult to use, the LLI procedures are useful for intricate applications that require extra data communications control.

Section 12. Using the BMULTI Suspend/Resume Filter

This section explains how to install and deinstall the BMULTI Suspend/Resume filter system service. It also describes the conditions and constraints necessary for using the filter. Procedure calls for including Suspend/Resume support in user–written applications can be found in Section 11, "Low–Level Procedural Interface."

Appendix A. Status Codes and Troubleshooting

This appendix lists the status codes generated by BMULTI and solutions for various error conditions.

Appendix B. USASCII Code Charts

This appendix showa the USASCII codes, with the special allocation of codes for the BMULTI protocol.

Appendix C. CTOS Request Codes for BMULTI

This appendix lists the Request Codes for BMULTI.

Appendix D. Examples of Shift–In and Shift–Out Action

This appendix shows examples to clarify the actions occurring in the BMULTI transmit and receive modes.

Appendix E. Translation Tables

This appendix explains and lists the tables used for character and language translation.

Appendix F. BMULTI Configuration Worksheet

This appendix provides a worksheet so that system administrators can record BMULTI configuration parameters.

Appendix G. Sample Programs

This appendix lists sample programs that use BMULTI procedural interfaces, written in Pascal.

A glossary and an index follow Appendix G. Additional information can be found in the Release Notes, located on the product disk.

Related Product Information

You may find the following documents helpful in performing the tasks described in this guide. These documents (or their replacements) are published by Unisys Corporation and can be ordered through Corporate Software and Publication Operations.

Product Documentation

In addition to the *CTOS BMULTI Operations and Programming Guide,* the following printed product information is packaged with release 10.1 of this product:

CTOS BMULTI Agent Operations Guide (Release 10.1)
CTOS BMULTI/X.25 Operations Guide (Release 1.1)
CTOS Printer Pass–Through (PPT) Operations Guide (Release 2.2)

CTOS Documentation

CTOS programmers and operators who want details on CTOS, CTOS procedural interfaces, Standard Software commands, Executive features, and status codes may find the following manuals useful:

BTOS II Standard Software Operations Guide
BTOS II System Reference Manual, 2 vols.
BTOS II System Procedural Interface Reference Manual, 2 vols.
BTOS II System Status Codes Reference Manual

CTOS Executive User's Guide
CTOS Executive Reference
CTOS Status Codes Reference Manual

Notation Conventions

This guide uses the following typographical conventions:

Words that you are to type and that appear on the screen are shown in *italic.*

Parameter names are shown in **bold.**

Keys are shown in uppercase letters, for example, press the GO key.

When two keys are to be used together for an operation, they are hyphenated. For example, SHIFT–RETURN means that while you hold down the SHIFT key, you press the RETURN key.

Contents

Section 3. Software Installation

Section 4. Configuring and Installing BMULTI

Section 5. The Status Monitor

Section 6. The Line Monitor

Section 9. Protocol Description

Section 10. High–Level Procedural Interface

Section 11. Low–Level Procedural Interface

Section 12. Using the BMULTI Suspend/Resume Filter

Appendix E. Translation Tables

Tables

Figures

Section 1
Overview

Product Description

CTOS BMULTI communications service allows CTOS workstations and shared resource processors to communicate with Unisys A-Series and V-Series mainframe systems using the BMULTI multipoint poll/select protocol. It enables applications such as BTE, File Transfer, DTS, and others to communicate with A-Series and V-Series host systems. This product also provides a programmatic interface so you can write applications to use the BMULTI communication service.

Features

Configuration File Editor

The BMULTI Configuration File Editor enables you to customize BMULTI for your system. Using the Configuration File Editor, you can edit different copies of BMULTI configuration files. You can also dynamically reconfigure BMULTI, changing some of the operational parameters of BMULTI after it is installed and running.

Status Monitor

The BMULTI Status Monitor is used to determine which applications are using BMULTI, and how busy they are. You may also use the status monitor to purge an address (unless you are running the monitor from a remote BNet node).

Line Monitor

The BMULTI Line Monitor can be used to show all data passing through BMULTI. To differentiate between incoming and outgoing data, incoming data is displayed in normal video, and outgoing data is displayed in reverse.

Using the line monitor, you can:

- "Freeze" the screen display while data is being received
- Store up to 60 KB of data in buffer memory
- Save any portion of buffered data to a disk file
- Scroll through buffered data using the page and scroll keys, or the "jump" function
- Load and examine data that was saved previously by the line monitor

Procedural Interfaces

BMULTI features High–Level and Low–Level procedural interfaces (HLI and LLI, respectively) for use in applications programming. The procedural interfaces allow user–written applications to use BMULTI data communications services.

The High–Level Interface is the easiest to use and provides functions sufficient for most programming needs. The Low–Level Interface is more difficult to use, but provides a greater degree of control.

Multiple Gateways Server

BMULTI supports the use of CTOS Multiple Gateways Server (MGS), which allows you to install multiple copies of BMULTI on a single workstation.

An individual copy of BMULTI running under MGS is identified by a Service ID, a parameter in the configuration file for each copy of BMULTI. Each copy of BMULTI must have a unique Service ID, taking the form <*ServiceID*>, with the name between one and twelve characters, enclosed in angle brackets.

BMULTI Agent

BMULTI Agent provides an asynchronous message–level interface between the BTOS Message Control System (MCS) and the BMULTI gateway. The interface between the BMULTI Agent and gateway allows you to write applications under MCS without having detailed knowledge of the poll/select protocol requirements. This optional system service can be selected during Software Installation. For more information, refer to the *CTOS BMULTI Agent Operations Guide*.

BMULTI/X.25 Filter

The BMULTI/X.25 filter enables BMULTI applications to communicate with Unisys mainframe systems over an X.25 Public Data Network (PDN). This optional system service can be selected during Software Installation. To use the BMULTI/X.25 filter, you must also have the BTOS X.25 Gateway installed as a system service. For more information, refer to the *CTOS BMULTI/X.25 Operations Guide.*

BNet Routable

BMULTI utilities and procedural interfaces support access from a remote node through BNet.

Some BMULTI utility functions, however, cannot be used over BNet; for example, with the status monitor, you cannot purge an address or channel at a remote node, but you can monitor BMULTI data communications activity at the remote node.

Additional Features

BMULTI supports:

- Configuration to answer polls of up to 64 addresses; the number of active addresses you should use depends on the applications and on the response time you need
- Messages of up to 4096 bytes (including protocol handling characters)
- Configuration for few users, reducing the server size in system memory
- Asynchronous and synchronous transmission at rates of 110 to 64,000 bps; the specific maximum rate available depends on your system, modem, and software configuration
- Normal poll, normal select, group poll, group select, fast select, broadcast select, and multipoint contention
- Several transmission numbering options
- Extended character sets; BMULTI uses the Shift-In/Shift-Out techniques of code extension to allow transmission of extended character sets

- User configuration of delays for Clear–to–Send, Transmit–to–Receive, and Request–to–Send–Hold
- A font file for the line monitor that displays protocol characters as two small characters in a single character space
- Deinstallation of BMULTI on server and client workstations

New Features

In addition to the previously mentioned features, this release of CTOS BMULTI supports the following new features:

- BMULTI Configuration File Editor now displayes full–text error messages. In previous releases of BMULTI, when errors occurred during configuration, the Configuration File Editor displayed only error codes. Now, explanatory error messages are displayed in addition to the codes.
- BMULTI provides auto–dial and auto–answer capabilities through the CTOS Modem Service. Through the Modem Service, BMULTI can automatically call a host or answer host calls over dial–up phone lines (in both synchronous and asynchronous modes).
- BMULTI and all its related component products now support NLS (Native Language Support) message files.
- The BMULTI gateway can use the Direct Memory Access (DMA) Input/Output (I/O) hardware in the IDS and enhanced IDS modules and on the XE-530 GP/CI board. The DMA hardware is available on channel B on IDS and enhanced IDS modules and channels A through H on the XE-530 GP/CI board. DMA operation must always be used for X.21 connections, but is optional for RS232C connections. DMA allows for high–speed RS232C synchronous and asynchronous communications.

- The BMULTI gateway can communicate over an X.21 leased or circuit–switched line. To use X.21, you must use the IDS or enhanced IDS module port B, or the XE-530 GP/CI board ports G or H. To operate over switched X.21 lines, the BTOS X.21 Circuit Switching Service (CSS) is required (CSS is not required for leased line operations). X.21 switched line support is available only on the enhanced IDS module and the XE-530 GP/CI board. In addition, for the XE-530, a converter is required to change the 25–pin D–connector to a 15–pin D–connector.

Note: For X.21 service, BMULTI always uses DMA. All DMA configuration restrictions and error situations apply to X.21 modes as well.

- The BMULTI Suspend/Resume Filter, a separately packaged component of the BMULTI product, allows different BMULTI applications to share the same BMULTI address, and therefore the same host session. For each session with the host, only one application can be active on an address at a time, but the Suspend/Resume filter allows applications to suspend their control of the session address so that another application can assume control. The filter installs as a separate BMULTI system service and transfers session control to different applications through session suspending and resuming.

  Some Unisys BMULTI software products (such as BTE 3.1) already support the BMULTI Suspend/Resume filter. User–written applications can make use of the filter through new Low–Level Interface (LLI) procedure calls. Refer to Section 12, "Using the BMULTI Suspend/Resume Filter," and Section 11, "Low–Level Procedural Interface," for more information.

  The Suspend/Resume filter is an optional system service, which can be selected during Software Installation.

- Some error codes have been added or redefined. Refer to Appendix A, "Status Codes and Troubleshooting," for more information.

- Installation Manager is supported.

Tasks and Commands

Task	Command	Section
Determine the system requirements for using BMULTI	Not applicable	2
Install the BMULTI software on a CTOS system	Software Installation/ XESoftware Installation or Installation Manager	3
Configure and customize BMULTI for your system	Configure BMULTI	4
Install or deinstall BMULTI as a system service	Install BMULTI Gateway or Deinstall BMULTI Gateway	4
Examine the status of BMULTI applications, or to purge an address (from the local node only)	BMULTI Status	5
Examine all data passing through BMULTI	Monitor BMULTI	6
Understand the functions and protocols of BMULTI	Not applicable	7, 8, 9
Write programs using the BMULTI High–Level Procedural Interface	Not applicable	10
Write programs using the BMULTI Low–Level Procedural Interface	Not applicable	11
Use the BMULTI Suspend/Resume filter; install or deinstall the Suspend/Resume filter system service	Install BMULTI S/R Filter or Deinstall BMULTI S/R Filter	12
Diagnose and correct error conditions	Not applicable	Appendix A

Section 2
System Requirements

This section describes the hardware, software and other system requirements for using CTOS BMULTI.

Hardware

BMULTI can be used on CTOS systems that use any of these processors:

- Real mode (186 processors)
- Protected mode (286 and 386 processors)
- Shared resource processors (XE520 and XE–530)

Notes: *BMULTI can only be stored on the shared resource processors, that is, its files can be physically located on the hard disk associated with the shared resource processor. The system service can be installed on CTOS workstations or shared resource processors. To operate the BMULTI Configurator, Status Monitor, or Line monitor, BMULTI requires a CTOS workstation.*

For X.21 operation, you need to use channel B on the IDS or enhanced IDS module, or channel G or H on the XE-530 GP/CI board.

XE520 shared resource processors must have a memory expansion board to use BMULTI.

Optional Hardware

In addition to the previously listed CTOS system processors, the following optional hardware can also be used with BMULTI:

- IDS or enhanced IDS module (B25–IDS or B25–ID2)
- Data Communications Expansion module (DCX)
- B24–TWS peripherals, including when attached to a shared resource processor
- Any CTOS systems printer
- AGI (advanced graphics processor)
- Enhanced Video
- VGA hardware
- EXP (extended processor)

Software

The following software is required to use this product:

Software	Release Level
BTOS II (for CTOS workstations)	3.0 or higher
CTOS/XE (for shared resource processors)	3.0 or higher
XEBTOS (for shared resource processors)	7.0 or higher
IDMSS (for IDS and enhanced IDS modules)	4.0 or higher

Optional Software

- Multiple Gateways Server (MGS) 1.0 or higher
- Modem Service 5.0 or higher
- X.21 Circuit Switching Service (CSS) 1.2 or higher (required for X.21 switched–line operation)
- BNet 2.1.3 or higher
- Context Manager II 1.2 or higher
- BTE 3.1 or higher

Disk Usage

Table 2–1 shows the approximate disk space required by each of the BMULTI files copied to your hard disk when you install BMULTI.

During the installation process, the installation of some BMULTI components is optional. Specifically, you can choose whether or not to install the BMULTI poll/select gateway, the BMULTI/X.25 gateway, the BMULTI Agent, the Line Monitor, and/or the Suspend/Resume filter. Choosing to install a subset of the components reduces the amount of disk space required.

Table 2–1. BMULTI Disk Space Requirements

File	Disk Size (Sectors)
BmAgentMsgFile.bin	8 sectors
BmConfigForm.lib	31 sectors
BmFileEdit.run	232 sectors
BmHelp.txt	5 sectors
BmLineMonitor.run	128 sectors
BmMonitorFont.AGP	22 sectors
BmMonitorFont.B25	16 sectors
BmMonitorFont.B27	16 sectors
BmMonitorFont.VE1	32 sectors
BmMsgFile.bin	13 sectors
BmSRF.run	42 sectors
BmSRFDin.run	29 sectors
BmSRFMsg.bin	2 sectors
BmStatus.run	100 sectors
BmStatusForm.lib	5 sectors
Bmulti.lib	40 sectors
Bmulti.run	121 sectors

continued

Table 2–1. BMULTI Disk Space Requirements (continued)

File	Disk Size (Sectors)
BmultiAgent.run	99 sectors
BmultiAgentConfig.run	124 sectors
BmultiAgentForm.lib	14 sectors
BMultiConfig.sys	4 sectors
BMultiXlat.sys	6 sectors
BmUnZip.run	25 sectors
BmX25.run	116 sectors
BmX25Config.sys	2 sectors
BmX25MsgFile.bin	6 sectors
BmX25UnZip.run	30 sectors
BmX25Xlat.sys	6 sectors
BmX25Zip.run	104 sectors
BmZip.run	78 sectors

The values provided are for estimation purposes only. The RAM usage for the BMULTI system services depends on your specific configuration. As for the disk size values, the software installation process itself will require more disk space than indicated due to the creation of temporary files.

System Configuration

To use BMULTI, your CTOS workstation or shared resource processor must be connected to a Unisys A-Series or V-Series mainframe. Typically, the connection is through a direct line, a leased line, or a switched telephone network. Additional hardware is necessary to connect a CTOS workstation or shared resource processor to any of these lines.

Modems and Modem Option Settings

In many systems, modem options are dictated by the conditions of the line to which the CTOS workstation or shared resource processor is connected. If using Modem Service 5.0 or higher, follow the instructions in the *CTOS Modem Service Operations Guide* to set up your modem. Otherwise, you should use these modem option settings:

- Transmitter internally timed
- Four–wire operation
- Switched carrier
- No new synchronization

You may use either synchronous or asynchronous two– or four–wire modems with BMULTI.

A double–male RS232C extension cable must be used to connect the CTOS system to the modem. It should be a straight–through terminal–to–modem cable, rather than the crossover (null modem) type.

Data Communications Adapter (DCA)

If a CTOS system is to be connected to a modem line with other Unisys terminals, a DCA is required to place the CTOS workstation anywhere other than last in the concatenation string.

RS232C signals used in operation are shown in Table 2–2. Those used in synchronous operation only are designated as such.

Table 2–2. RS232C Signals in Operation

Pin Number	Signal Name
1	Protective Ground
2	Transmit Data
3	Receive Data
4	Request to Send (RTS)
5	Clear to Send (CTS)
6	Data Set Ready
7	Signal Ground
8	Data Carrier Detect (Not used on CTOS workstations)
9–13	Not used
14	Block Downstream CTS (Not used on shared resource processors)
15	Transmit Clock (Synchronous operation only)
16	Sense Downstream RTS (Not used on shared resource processors)
17	Receive Clock (Synchronous operation only)
18–19	Not used
20	Data Terminal Ready

Migration and Coexistence

Using Old BMULTI Configuration Files

Any configuration file created with previous versions of BMULTI must be updated to the current version. To do this, open the file using the BMULTI Configuration File Editor for the older version of BMULTI. Use a screen–capture print utility or write down the information on the various configuration menus to record the parameter and settings you have been using. Create new files using the default BMULTI configuration file as a template, and use the Configuration File Editor for the current version of BMULTI to fill in the required information.

Adapting Applications Written for Early Versions of BMULTI

The current system services do not support any pre–6.0 versions of the BMULTI libraries. Any application using these library versions must be rewritten, recompiled, and relinked using the current BMULTI library. In addition, any application linked with the current library also requires the 6.0 or higher version of the BMULTI system service.

To use BNet and/or MGS capabilities, you need to enter the node name and/or Service ID in the host name field of the BmOpenII procedure call (see Section 11, "Low–Level Procedural Interface"). To make use of the BMULTI Suspend/Resume capabilities, you must use the BmResume call instead of BmOpenII.

Host applications do not require reprogramming to interface with this product.

Multiple Gateways Server (MGS) Compatibility

BMULTI is compatible with BTOS Multiple Gateways Server (MGS). Installation under MGS allows you to install multiple copies of BMULTI on the same CTOS system.

To function with BMULTI, MGS needs to be installed with the following constraints for its configuration parameters:

Parameter	**Minimum Value**
Minimum number of requests	13
Minimum number of outstanding requests	(2 x *#Stations*) + 1
Minimum number of file handles	*#Stations*

The value *#Stations* represents the total number of stations for all the copies of BMULTI to be run under MGS.

The minimums described are only the lower limits; MGS can be installed with parameters larger than those listed, and will be larger if other types of gateways are also operating with MGS. The number of stations can be found by using the BMULTI Configuration File Editor to open and examine the BMULTI configuration files.

To operate with MGS, the BMULTI configuration file(s) must be modified to include a *ServiceID*. A Service ID is a name of up to 12 alphanumeric characters (no spaces), enclosed by angle brackets (<>).

The Service ID is used by MGS to identify different copies of BMULTI running on the same system. The system administrator should keep track of all Service IDs assigned.

When BMULTI is installed as a system service under MGS, the Service ID is assigned as the partition name.

MGS Support for Previous Versions of BMULTI

If you are using applications linked with a pre–9.0 BMULTI High–Level Interface (HLI) library, the node name field must be handled specially for applications using Service IDs.

To accommodate Service IDs in applications using the "SelectBMULTI" procedure in pre–9.0 libraries, the node name field must contain a valid {*NodeName*}; if there is no node name, you can enter {MASTER} or {LOCAL}. An additional closing brace character is required after the Service ID for compatibility.

Use the following convention:

{*NodeName*}<*ServiceID*>}

{MASTER}<*ServiceID*>}

{LOCAL}<*ServiceID*>}

These conventions are not required for applications linked with BMULTI libraries included in releases of BMULTI 9.0 or higher.

Context Manager

The BMULTI status and line monitors can be used with the CTOS Context Manager II.

The following lists information you may need when configuring Context Manager II to run this product. For more information, refer to the *BTOS Context Manager II Installation and Configuration Guide.*

Context Manager Configuration for the Status Monitor

Run file name	[Sys]<Sys>BmStatus.run
Runs in protected mode	Yes
Memory required	46 KB
Able to be swapped	Yes
Needs executive screen	No

Context Manager Configuration for the Line Monitor

Run file name	[Sys]<Sys>BmLineMonitor.run
Runs in protected mode	Yes
Memory required	72 KB
Able to be swapped	Yes
Needs executive screen	No

Section 3
Software Installation

This section explains how to copy the BMULTI software onto a workstation or a shared resource processor. It includes instructions for using two methods of installation.

The first method uses the software installation commands (Software Installation and XESoftware Installation), which are available with any of the supported operating systems (see Section 2, "System Requirements," for a list of supported operating systems). The second method uses the Installation Manager command, which is available with BTOS II 3.2, CTOS/XE 3.0, and CTOS Standard Software 12.0.

For a more detailed explanation of the software installation commands, consult the *BTOS II Standard Software Operations Guide.* For a more detailed explanation of the Installation Manager command, consult the *CTOS Executive Reference Manual.*

Using Software Installation

You can use the software installation commands to copy software from floppy disks onto workstations or shared resource processors.

Workstations

To install BMULTI, you must make the software available for your workstation. Use the following procedure if you plan to use BMULTI on a server, client, or stand–alone workstation.

Note: *BMULTI is delivered on three disks. Start with the disk labeled B25BMU-1. You will be prompted when to use the second disk. The third disk contains the CTOS Printer Pass–Through (PPT) software product. Refer to the* CTOS Printer Pass–Through (PPT) Operations Guide *for information on installing and using PPT.*

1. If you are installing BMULTI on a server workstation, power down all client workstations booted from the server.

2. Insert the disk labeled *B25BMU-1* into floppy disk drive [F0].
3. At the Executive Command line, type *Software Installation* and press the GO key.
4. During installation, you will receive a prompt asking you if you want to read or print the Release Notes. The Release Notes contain last–minute information about the product and updates to this guide, if any. Follow the on–screen instructions for reading or printing this file.
5. During the installation, you will be offered prompts to select optional BMULTI components. These include the BMULTI gateway, the BMULTI Agent, the BMULTI/X.25 gateway, the Line Monitor, and the Suspend/Resume filter. Information on the BMULTI Agent and BMULTI/X.25 can be found in the *CTOS BMULTI Agent Operations Guide* and the *CTOS BMULTI/X.25 Operations Guide*, respectively. The BMULTI Line Monitor and Suspend/Resume filter are described in Sections 6 and 12 of this guide, respectively.
6. Follow the prompts that appear on the screen to complete the installation. The necessary files are copied, the request files are updated, and the commands to run BMULTI are created.
7. Remove the last disk and keep the disk set in a secure place.
8. Reboot the workstation or server, and power up any client workstations that were powered down before installing BMULTI.

Shared Resource Processor Systems

When a shared resource processor is the server, you copy BMULTI onto the shared resource processor's hard disk from a client workstation. If you have an XE520 shared resource processor running the XEBTOS operating system, use the XESoftware Installation command. If you have either an XE520 or XE-530 running the CTOS/XE operating system, use the Software Installation command.

XESoftware Installation

Use the following procedure if you plan to use BMULTI on an XE520 system running XEBTOS.

Note: *BMULTI is delivered on three disks. Start with the disk labeled B25BMU-1. You will be prompted when to use the second disk. The third disk contains the CTOS Printer Pass–Through (PPT) software product. Refer to the* CTOS Printer Pass–Through (PPT) Operations Guide *for information on installing and using PPT.*

1. Power down all client workstations booted from the shared resource processor, except for the one at which you are performing the installation.
2. Insert the disk labeled *B25BMU-1* into floppy disk drive [F0] at the workstation connected as a client to the shared resource processor.
3. At the Executive command line, type *XESoftware Installation* and press the GO key.
4. You will receive a prompt asking you if you want to read or print the Release Notes. The Release Notes contain last–minute information about the product and updates to this guide, if any. Follow the on–screen instructions for reading or printing this file.
5. During the installation, you will be offered prompts to select optional BMULTI components. These include the BMULTI gateway, the BMULTI Agent, the BMULTI/X.25 gateway, the Line Monitor, and the Suspend/Resume filter. Information on the BMULTI Agent and BMULTI/X.25 can be found in the *CTOS BMULTI Agent Operations Guide* and the *CTOS BMULTI/X.25 Operations Guide*, respectively. The BMULTI Line Monitor and Suspend/Resume filter are described in Sections 6 and 12 of this guide, respectively.
6. Follow the prompts that appear on the screen to complete the installation. The necessary files are copied, the request files are updated, and the commands to run BMULTI are created.
7. Remove the last disk and keep the disk set in a secure place.
8. Reboot the shared resource processor, and power up any client workstations that were powered down before installing BMULTI.

Software Installation

Use the following procedure if you plan to use BMULTI on an XE520 or XE-530 runing CTOS/XE.

Note: *BMULTI is delivered on three disks. Start with the disk labeled B25BMU-1. You will be prompted when to use the second disk. The third disk contains the CTOS Printer Pass–Through (PPT) software product. Refer to the* CTOS Printer Pass–Through (PPT) Operations Guide *for information on installing and using PPT.*

1. Power down all client workstations booted from the shared resource processor, except for the one at which you are performing the installation.
2. At a client workstation, boot the workstation from the shared resource processor. (When you do this, the [Sys] volume should refer to the hard disk on the shared resource processor.

   Note: *Make sure you are not using Cluster View (a utility that allows the XE board to be controlled from a client workstation). If you are using Cluster View, the system will not recognize the floppy disk drive.*
3. Insert the disk labeled *B25BMU-1* into floppy disk drive [F0] at the client workstation.
4. At the Executive command line, type *Software Installation* and press the RETURN key. The following form appears:

   ```
   Software Installation
     [Cmd File]
     [Files to]
     [Install file]
   ```

   The default for the **[Cmd File]** parameter is "[Sys]<Sys>Sys.Cmds." The default for the **[Files to]** parameter is "[Sys]<Sys>." You may need to change these values to the [!Sys]<Sys> directory or another directory containing a command file.

   Change the values, if necessary, and press the GO key.
5. You will receive a prompt asking you if you want to read or print the Release Notes. The Release Notes contain last–minute information about the product and updates to this guide, if any. Follow the on–screen instructions for reading or printing this file.

6. During the installation, you will be offered prompts to select optional BMULTI components. These include the BMULTI gateway, the BMULTI Agent, the BMULTI/X.25 gateway, the Line Monitor, and the Suspend/Resume filter. Information on the BMULTI Agent and BMULTI/X.25 can be found in the *CTOS BMULTI Agent Operations Guide* and the *CTOS BMULTI/X.25 Operations Guide*, respectively. The BMULTI Line Monitor and Suspend/Resume filter are described in Sections 6 and 12 of this guide, respectively.
7. Follow the prompts that appear on the screen to complete the installation. The necessary files are copied, the request files are updated, and the commands to run BMULTI are created.
8. Remove the last disk and keep the disk set in a secure place.
9. Reboot the shared resource processor, and power up any client workstations that were powered down before installing BMULTI.

Using Installation Manager

If you are using a version of the CTOS operating system with the Installation Manager command, you can use the Installation Manager command to copy software from floppy disks or QIC tapes onto workstations or shared resource processors. You can also use this command to ensure that the required software is installed, to install software from a server to a client workstation, and to remove software when you no longer need it at a particular location.

Viewing Installed Software

Before you install the new software, view the software already installed on your workstation or shared resource processor to be sure you have all the software required to use BMULTI. (Refer to Section 2, "System Requirements," for a description of required software.) Use the following procedure:

1. At the Executive Command line, type *Installation Manager* and press the RETURN key. The following form appears:

```
Installation Manager
  [Noninteractive?]
```

2. Type *No* and press the GO key. The Software Operations menu appears.
3. Select the *Show Installed Software* option to view the software already installed by Installation Manager. Press the GO key.
4. Select *Private Software* to view software installed on the local hard disk; or select *Public Software* to view the software installed on the server. Press the GO key. A list of currently installed software applications appears.
5. Press the CANCEL key twice to return to the Software Operations menu.
6. Install any software required to run BMULTI.

Installing Software

Before you can install BMULTI, you must copy the software onto your workstation, server, or shared resource processor system.

Installation Manager works the same way for workstations and shared resource processors.

Note: *BMULTI is delivered on three disks. Start with the disk labeled B25BMU-1. You will be prompted when to use the second disk. The third disk contains the CTOS Printer Pass–Through (PPT) software product. Refer to the* CTOS Printer Pass–Through (PPT) Operations Guide *for information on installing and using PPT.*

1. If you installing BMULTI on a server, power down any client workstations booted from the server.
2. Insert the disk labeled *B25BMU-1* into floppy disk drive [F0] at the workstation.
3. At the Executive Command line, type *Installation Manager* and press the RETURN key. The following form appears:

```
Installation Manager
  [Noninteractive?]
```

4. Type *No* (for interactive mode) and press the GO key. The Software Operations menu appears.

   Note: *If you leave this parameter blank, the default is* NO. *Interactive mode lets you select options and read the Release Notes, and should always be used when copying from floppy disks. Noninteractive mode installs the software without pausing for user input.*

5. Select the *Install New Software* option and press the GO key. The Installation Media menu appears.

6. Select the medium you are using to install BMULTI, *Floppy Installation*, then press the GO key. The Installation menu appears.

7. You can continue installation with the default installation parameters by pressing the GO key, or you can examine/change the installation defaults by selecting the *Examine / Change Defaults* option and then pressing the GO key.

   Note: *Unless you pre–configured your installation parameters through Installation Manager, all the defaults are* NO. *This means the software will be installed on only your local workstation, and any currently installed version of the software will be overwritten. You must change the* ***Public Installation*** *parameter to* Yes *to install the software on a server or shared resource processor.*

8. If you chose the *Examine / Change Defaults* option:

   a. Decide whether you want the software installed on the server or only on your local workstation. Installing the software on the server allows all the client workstations in the cluster to access the software without using their own disk space. However, to use BMULTI from the server, you need to update the Request.Sys files on the client workstations.

      If you want the software installed on the server, type *Yes* in the **Public Installation** parameter.

      If you want the software installed on your local workstation, type *No* in the **Public Installation** parameter.

b. Change any other defaults, if necessary, and press the GO key. The User Name menu appears.

c. Change the default user name, if necessary, and press the GO key. The Command File for appears.

d. Change the Command File name if necessary, and press the GO key.

9. You will receive a prompt asking you if you want to read or print the Release Notes. The Release Notes contain last–minute information about the product and updates to this guide, if any. Follow the on–screen instructions for reading or printing this file.

10. During the installation, you will be offered prompts to select optional BMULTI components. These include the BMULTI gateway, the BMULTI Agent, the BMULTI/X.25 gateway, the Line Monitor, and the Suspend/Resume filter. You are also offered the option to just update the Request.Sys file. Information on the BMULTI Agent and BMULTI/X.25 can be found in the *CTOS BMULTI Agent Operations Guide* and the *CTOS BMULTI/X.25 Operations Guide*, respectively. The BMULTI Line Monitor and Suspend/Resume filter are described in Sections 6 and 12 of this guide, respectively.

11. Follow the prompts that appear on the screen to complete the installation. The necessary files are copied, the request files are updated, and the commands to run BMULTI are created.

12. When the installation is finished, a message appears stating that the installation has completed successfully and asks if you want to reboot the system. Remove the last disk and keep the disk set in a secure place.

13. Reboot the workstation or server and power up any client workstations that were powered down before installing BMULTI.

Files Copied to Hard Disk

If you choose during software installation to install the BMULTI gateway, the files created on the hard disk during BMULTI software installation are:

File	Description
<Sys>BmConfigForm.lib	The BMULTI configuration file forms
<Sys>BmFileEdit.run	Edits the BMULTI configuration files
<Sys>BmMsgFile.bin	The BMULTI message file
<Sys>BmStatus.run	Runs the BMULTI status monitor utility
<Sys>BmStatusForm.lib	The BMULTI status monitor forms
<Sys>Bmulti.lib	The library containing the BMULTI High-Level and Low-Level procedural interfaces
<Sys>Bmulti.run	The BMULTI system service
<Sys>BMulticonfig.sys	The default BMULTI configuration file
<Sys>BMultiXlat.sys	The BMULTI translation table file
<Sys>BmUnZip.run	Deinstalls the BMULTI system service
<Sys>BmZip.run	Installs the BMULTI system service

If you choose during software installation to install the BMULTI Line Monitor, the following files are created on your hard disk:

File	Description
<Sys>BmHelp.txt	The BMULTI status monitor help file
<Sys>BmLineMonitor.Run	Runs the BMULTI line monitor utility
<Sys>BmMonitorFont.AGP	The font file for advanced graphics systems
<Sys>BmMonitorFont.B27	The font file for B 27 systems
<Sys>BmMonitorFont.B25	The font file for B 26, B 28, B 38, and B 39 systems
<Sys>BmMonitorFont.VE1	The font file for B 24 systems

If you choose during software installation to install the BMULTI Agent, the following additional files are created on your hard disk:

File	Description
<Sys>BmAgentMsgFile.bin	The BMULTI Agent message file
<Sys>BmultiAgent.run	The BMULTI Agent system service
<Sys>BmultiAgentConfig.run	Runs the BMULTI Agent configurator utility
<Sys>BmultiAgentForm.Lib	The BMULTI Agent forms library

If you choose during software installation to install the BMULTI Suspend/Resume filter, the following additional files are created on your hard disk:

File	Description
<Sys>BmSRF.run	Installs the BMULTI Suspend/Resume system service
<Sys>BmSRFDin.run	Deinstalls the BMULTI Suspend/Resume system service
<Sys>BmSRFMsg.bin	The BMULTI Suspend/Resume message file

If you choose during software installation to install the BMULTI/X.25 gateway, the following additional files are created on your hard disk:

File	Description
<Sys>BmX25.Run	The BMULTI/X.25 system service
<Sys>BmX25Config.Sys	The BMULTI X.25 configuration file
<Sys>BmX25MsgFile.bin	The BMULTI/X.25 message file
<Sys>BmX25Unzip.Run	Deinstalls the BMULTI/X.25 system service
<Sys>BmX25Xlat.sys	The BMULTI/X.25 translation file
<Sys>BmX25Zip.Run	Installs the BMULTI/X.25 system service

Commands Created

The following commands are created during software installation:

Command	Description
Install BMULTI Gateway	Installs BMULTI as a system service (this command is available only if the BMULTI gateway is selected during software installation)
Deinstall BMULTI Gateway	Deinstalls BMULTI as a system service (this command is available only if the BMULTI gateway is selected during software installation)
Configure BMULTI	Runs the BMULTI configurator, which allows you to edit BMULTI configuration files (this command is available only if the BMULTI gateway is selected during software installation)
BMULTI Status	Runs the BMULTI status monitor utility (this command is available only if the BMULTI gateway is selected during software installation)
Install BMULTI S/R Filter	Installs the BMULTI Suspend/Resume filter (this command is available only if the BMULTI S/R filter is selected during software installation)
Deinstall BMULTI S/R Filter	Deinstalls the BMULTI Suspend/Resume filter (this command is available only if the BMULTI S/R filter is selected during software installation)
Monitor BMULTI	Runs the BMULTI line monitor utility (this command is available only if the line monitor is selected during software installation)
Install BMX25	Installs the BMULTI/X.25 filter as a system service (this command is available only if BMULTI/X25 is selected during software installation)
Deinstall BMX25	Deinstalls the BMULTI/X25 gateway as a system service (this command is available only if BMULTI/X25 is selected during software installation)
BMULTI Agent Configurator	Runs BMULTI Agent configuration utility (this command is available only if the BMULTI Agent is selected during software installation)
Install BMULTI Agent	Installs the BMULTI Agent as a system service (this command is available only if the BMULTI Agent is selected during software installation)

For information about the BMULTI/X.25 gateway, refer to the *CTOS BMULTI/X25 Operations Guide*. For more information about the BMULTI Agent, a service that acts as a bridge between BMULTI and the CTOS Message Control System (MCS), refer to the *CTOS BMULTI Agent Operations Guide*.

Removing BMULTI From Your System

You can permanently remove software from your system to allocate space for other software or to use your copy of BMULTI on another system without infringing on copyright or license agreements. Depending on your operating system and hardware, there are several ways to remove your software.

Using Installation Manager to Remove Files and Commands

If you have BTOS II 3.2, CTOS/XE 3.0, or CTOS Standard Software 12.0 installed on your system, you can use the Installation Manager command to remove software that was installed by Installation Manager. Use the following procedure to remove software:

1. At the Executive Command line, type *Installation Manager* and press the GO key. The Software Operations menu appears.
2. Select the *Remove Software* option and press the GO key.
3. Select either *Public Software* or *Private Software* and press the GO key. A list of currently installed software applications appears.
4. Select the software package you wish to remove and press the GO key.

   Installation Manager deletes all related files and removes the associated commands.

Using the Delete Command to Remove Files

You can use the **Delete** command to remove all product related files from your system.

1. At the Executive command line, type *Delete* and press the RETURN key. The following form appears:

```
Delete
  File list
  [Confirm each?]
```

2. For the File list parameter, enter the files listed in the previous section, "Files Copied to Hard Disk." Press the GO key. You may need to execute this command several times to remove all the files because only so many files will fit on the line at a time, separated by spaces. You may use the wildcard characters (* and ?) to shorten the process, but use caution so that you do not accidentally delete files unrelated to BMULTI.

Using the Command File Editor to Remove Commands

You can use the Command File Editor to remove all product related commands from your system. The Command File Editor also allows you to add and modify commands.

1. At the Executive command line, type *Command File Editor* and press the RETURN key.
2. Move the cursor to select the BMULTI commands listed previously in this section under "Commands Created." For each one, press the F3 key (Remove), and then press the GO key to confirm removing the command.
3. Press the FINISH key, then the GO key to save the changes and exit the Command File Editor.

Removing Individual Commands

To remove individual commands from your system, you can use **Remove Command**.

1. At the Executive command line, type *Remove Command* and press the RETURN key.
2. For the **Old command name** parameter, type the name of the BMULTI command to be removed. If necessary, type the name of the command file, including the directory path, on the next parameter line. Press the GO key.
3. The command will be removed from the command file.
4. Repeat this procedure as needed to remove the other BMULTI commands.

Section 4
Configuring and Installing BMULTI

This section describes how to configure CTOS BMULTI and install it as a system service on a CTOS workstation or shared resource processor.

When a system service is installed, it is loaded into the RAM memory of a CTOS system, where it handles BMULTI requests. For more information about the system services, refer to the *BTOS II Standard Software Operations Guide (SSOG).*

BMULTI Configuration File Editor

The BMULTI Configuration File Editor enables you to customize BMULTI for your system. It can be used to create or modify one or more copies of the BMULTI configuration file, which is used to determine how BMULTI operates.

The BMULTI configuration file parameters are described later in this section, and a BMULTI configuration worksheet is included in Appendix F.

Note: *The configuration file format has changed for this release. If you have a configuration file from an earlier version of BMULTI, you must copy the information to a new file based on the default configuration file provided with this version of BMULTI.*

To do this, you can use the older version of the BMULTI Configuration File Editor to open the configuration file, then copy the information using a screen–capture print utility or write down the information. Use the BMULTI Configuration File Editor for this release and a copy of the BMULTI default configuration file to re–record the information.

Attempting to use a configuration file for an earlier release of BMULTI can cause unpredictable results.

Entering and Exiting the BMULTI File Configuration Editor

1. Type *Configure BMULTI* on the command line and press the RETURN key. The following form is displayed:

```
Configure BMULTI
  [Configuration file]
```

2. The default configuration file is [Sys]<Sys>BMULTIConfig.Sys. If you want to create a file with a different name, type the file name and press the GO key.
3. To exit the Configuration File Editor, press the FINISH key. You will be asked to press the GO key to confirm the exit. Otherwise, you can press the CANCEL key to stay in the Editor.

The Configuration File Editor retrieves or creates the file, as needed, and displays it on the screen. Figure 4–1 shows the Configuration File Editor display.

Using the Configuration File Editor

Following are guidelines and instructions for using the Configuration File Editor:

- To move within a field, use the right–arrow and left–arrow keys.
- To move to the following field, press the TAB key, the NEXT key, the RETURN key, or the down–arrow key. To move to the previous field, use the up–arrow key.
- If you press an invalid key, the workstation beeps and displays an error message at the top of the screen: "Invalid cursor movement."
- If you enter an unacceptable value, the workstation beeps and prevents you from leaving the field until you correct the value. If you cannot find an acceptable value, enter the appropriate default value from the parameter descriptions later in this section.
- The **Channel** and **Bit Rate** parameters must be right–justified (flush against the right side of the field), with no spaces in the field.

The display screens, parameter fields, and keyboard functions are described below:

Top Line

The top line of the display shows the name and version level of the Configuration File Editor. Directly beneath is a line which lists the name of the file currently open (if any).

Parameter Fields

Under the file name are the parameter fields. You move among the parameter fields by using the arrow keys. The Configuration File Editor allows you to make changes only to parameters, but not to any of the parameter labels.

See "Configuration Parameters" in this section for definitions of the parameters and permissible values for them.

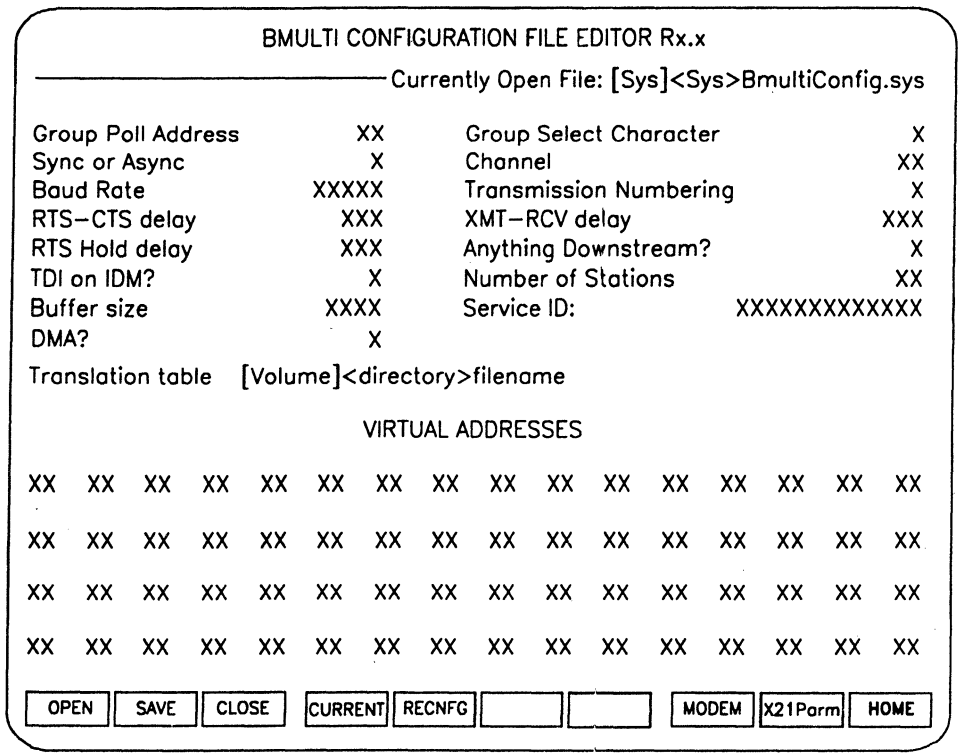

Figure 4–1. BMULTI Configuration File Editor Main Menu

Translation Table File Name

Below the parameter fields is a line that specifies the translation table file name. This file name represents the translation table that is loaded when BMULTI is installed. The table is used because the character sets for various languages supported by CTOS systems are not the same as those on the mainframes. As a consequence, some translation between characters sets is necessary both before transmitting and receiving. The default translation file is called BMULTIXlat.Sys and is the USA version. See Appendix E for more information about translation tables.

Virtual Addresses (VADs)

Next are the Virtual Addresses. In the default configuration file, the virtual addresses are left blank. Any ASCII two–character combinations may be used as virtual addresses.

Virtual addresses (VADs) are used with certain configurations (which are described below) to avoid excessive line timeouts and to improve the performance of terminals on the same line. A VAD acts as if it is an active address in the "Idle" state. The LLI requests BmOpenII or BmResume can be used to open an unused, previously defined VAD.

BMULTI searches for VADs after searching for active addresses. BMULTI always replies with EOT for polls and NAK for selects for each inactive VAD, meaning that no application is using that address.

VADs should be used only with unsophisticated Network Definition Languages (NDLs). These NDLs are typically ones used on CMS systems and on Unisys small systems. The NDLs used in these environments send poll sequences to inactive stations (or addresses) until a timeout or number of retries is exceeded. After either of these conditions becomes true, that station is removed from the poll sequence and can be reactivated only through the host operator's intervention. In this environment, VADs are particularly useful because they keep the line active, although idle.

Indiscriminate use of VADs on systems with sophisticated NDLs can cause a noticeable performance degradation on the data communications line. This is more noticeable when a large number of VADs are defined in the BMULTI configuration file.

Do not use VADs with the NDLs used on large systems (for example, B7900) and the A-Series mainframes. NDLs on these systems poll inactive addresses less frequently than active addresses, so the line is not overloaded with unnecessary polls and selects. If a station becomes active, the host eventually starts polling that terminal more frequently without any operator intervention.

Function Key Menu

At the bottom of the display is the function key menu. The function keys and their meanings are:

Function Key	Description
OPEN (F1)	Opens a configuration file for editing. When you press the OPEN key, the menu label is highlighted and you are prompted for the file name. Another field asks for the password (if any) for the configuration file.
SAVE (F2)	Saves the currently displayed parameter values into the open configuration file.
CLOSE (F3)	Closes the current configuration file, but does not save it. To save the parameter values to the configuration file, you must press the SAVE key (F2) *before* you press the CLOSE key.
CURRENT (F4)	Displays the parameters being used by BMULTI. When running under Multiple Gateways Server (MGS), you must specify the Service ID of the particular copy of BMULTI you want to examine. To make changes to these parameters, press the HOME key (F10). Then, to use the changed parameters press the RECNFG key (F5).

Function Key	Description
RECNFG (F5)	Reconfigures BMULTI using the parameters on the Configuration File Editor display. This function allows you to make changes to BMULTI configuration and implement them immediately, without having to deinstall and reinstall the BMULTI server. The RECNFG key can also be use for reconfiguration with different configuration files When using this function, BMULTI should be in an idle and stable state, and not all parameters are dynamically reconfigurable. You should exercise caution when using this function. When running under Multiple Gateways Server (MGS), you must specify the Service ID of the particular copy of BMULTI you want to reconfigure.
Modem (F8)	Enters the modem configuration menu. This menu provides configuration options for Modem Service capabilities. Through the CTOS Modem Service, BMULTI provides auto–dial/auto–answer capabilities on synchronous and asynchronous dial–up phone lines. Refer to the subsection, "Modem Service Configuration," later in this section, for more information.
X21Parm (F9)	Enters the X.21 parameter configuration menu. In this menu, the function keys are reassigned so that the HostInfo key (F8) activates the X.21 host information menu, the X21Parm key (F9) returns to the X.21 parameter menu, and the HOME key (F10) returns to the main BMULTI configuration editor menu. Refer to the subsection, "X.21 Configuration," later in this section, for more information.
HOME (F10)	Returns to editing the parameter fields; by default, this key is highlighted when you enter the BMULTI configuration editor. Once parameters are edited, you must reinstall BMULTI or press the RECNFG key (F5) to implement new parameters.

Dynamic Reconfiguration of BMULTI

Dynamic reconfiguration of BMULTI is a process by which you can change the operating parameters of the BMULTI system service as it runs in memory. You can perform dynamic reconfiguration only after BMULTI is installed as a system service. Also, the reconfigure function (RECNFG) in the Configuration File Editor does not work unless BMULTI is installed.

The parameter values entered during dynamic reconfiguration are not saved to disk in a configuration file. They exist only in RAM and are current only as long as BMULTI remains installed. You cannot save dynamically reconfigured values to a file.

CAUTION

In general, dynamic reconfiguration should only be performed when BMULTI is in a stable state with no connections or sessions active. For some parameters, such as X.21 configuration, DMA operation, or increasing buffer sizes, you should not (or, in some cases, you cannot) dynamically reconfigure them. Attempting to do may result in data loss or other unpredictable effects.

Some of the known restrictions for dynamic reconfiguration are listed below.

The only completely safe way to reconfigure BMULTI is to edit the configuration file, save it, then deinstall and reinstall the BMULTI gateway.

Following are some of the known restrictions for dynamic reconfiguration of BMULTI:

- You cannot increase the number of available stations.
- Buffer sizes cannot be dynamically reconfigured
- If DMA is set to *No* during BMULTI installation, it cannot be dynamically reconfigured to *Yes*.
- When using X.21 communications, you cannot dynamically change from leased–line to circuit–switched operation or vice versa.
- Dynamic reconfiguration from X.21 communications to Modem Service communications is not supported.
- When using Modem Service, you cannot dynamically change from synchronous to asynchronous communications or vice versa.

Using Configuration Files from Earlier Versions of BMULTI

If you have configuration files created with a BMULTI release level lower than the current level, the files must be updated before using them. If you attempt to use old configuration files, you may encounter unpredictable results.

To update your configuration files, you should open them using the Configuration File Editor from the previous release of BMULTI and record the information they define. Then make copies of the default BMULTI configuration file to replace the ones you are currently using, open them using the Configuration File Editor, make any new parameter changes or selections and save them.

BMULTI Configuration Parameters

The BMULTI configuration file contains the parameters used by BMULTI during its operation. To configure BMULTI for your environment, use the BMULTI Configuration File Editor to create and edit a configuration file. You can use the BMULTI status monitor to change some parameters but not all parameters after you have installed BMULTI; the status monitor is described in Section 5. See Appendix F for a BMULTI configuration worksheet.

The parameters described here are those in the main configuration editor menu. There are additional parameters displayed in the modem, X.21, and host information menus. Refer to the appropriate subsections for information about the associated parameters.

Main Menu Configuration Parameters

Group Poll Address (default = dv)

The group poll address to be used by all BMULTI stations. A client workstation or any workstation using BMULTI over BNet can have only one group address. This must be any two ASCII characters between 020h and 07Fh, as listed in Appendix B. If group poll is not used, any address not polled may be used.

Group Select Character (default = r)

The character recognized as the group select character to be used by BMULTI for all its stations. A client workstation or any workstation using BMULTI over BNet can have only one group select character. This must be any ASCII character between 020h and 07Fh, as listed in Appendix B. If group select is not used, you may use any character not already in use as a poll or select character.

Sync or Async (default = A)

Selects synchronous (*S*) or asynchronous (*A*) timing. In synchronous mode, the modem provides the clock signal; in asynchronous mode, the workstation supplies the clock signal. Always enter *A* for a TDI line.

Channel (default = 0B)

Selects the hardware channel (port) to be used for BMULTI:

CTOS Workstation (w/out DCX)	A, B (or 0A, 0B)
First DCX module (to right of CPU)	1A, 1B, 1C, 1D
Second DCX module	2A, 2B, 2C, 2D
Shared Resource Processor	A to H

The specific channels available will depend on your hardware configuration.

For BMULTI to take control of a specified channel, that channel must not be already under the control of another program. Make sure the channel you assign is not assigned to your system's spooler.

On a shared resource processor, no "ASYNC <channel number>" statement may refer to that channel in the appropriate configuration file *[Sys]<Sys>CPnn.cnf* for the cluster processor board and *[Sys]<Sys>TPnn.cnf* for the terminal processor board, where *nn* is the cluster or terminal processor number).

Bit Rate (default = 9600)

Transmission speed in bits per second (bps). For all systems, synchronous or asynchronous transmission can operate bit rates of 110, 150, 300, 600, 1000, 1200, 1800, 2000, 2400, 4800, or 9600 bps (the XE520 cannot use 110 or 150 bps). Maximums for individual systems, however, vary:

System	Sync	Async	DMA?
CTOS workstation	9600	19200	DMA not available
DCX module	9600	9600	DMA not available
IDS/enhanced IDS	9600	19200	Without DMA
(channel B)	64000	38400	Using DMA
XE520	9600	19200	DMA not available
XE-530	9600	19200	Without DMA
(GP/CI A–H)	64000	38400	Using DMA

As noted above, DMA operation requires the use of Port B on the IDS or enhanced IDS (B25–ID2) module, or channels A through H on the XE-530 GP/CI board.

When running multiple copies of BMULTI under MGS, the cumulative asynchronous bit rate must not exceed the maximum rate for the system.

Transmission Numbering (default = 0)

Each of these numbers represents a transmission numbering scheme:

Number	Transmission Numbering Scheme
0	No transmission numbers
1	Alternating zero and one
2	Alternating @ and A (TD830 compatibility)
3	*N* modulus 10 (0 through 9, wrapping around)
4	*N* modulus 100 (0 through 99, wrapping around)
5	*N* modulus 1000 (0 through 999, wrapping around)

See Section 9, "Protocol Description," for more information about transmission numbering.

RTS–CTS delay (default = 0)

The Clear–to–Send (CTS) delay in milliseconds (ms), between 0 and 255 ms. After you turn on Request–to–Send (RTS), BMULTI waits the amount of time set here before looking for CTS from the modem. If CTS is not on when the time expires, BMULTI waits until it goes on before it transmits.

XMT–RCV delay (default = 0)

The delay in milliseconds, between 0 and 255 ms, after turning off Request–to–Send (RTS), that BMULTI waits before examining incoming data. This delay is normally not zero when using Two–wire Direct Interface (TDI).

If you are using a Data Communications Adapter (DCA) for TDI to RS232C conversion, use the following recommended transmit–receive delay values:

Bit Rate	Delay Value
9600 or higher	1 ms (millisecond)
4800	2 ms
2400	4 ms
1200	8 ms
600	16 ms
300	32 ms

When using a DCA that eliminates echo, it may be possible to operate with a delay of zero. You may need to experiment to find the shortest delay required for the specific hardware installed.

For shared resource processors, a transmit–receive (XMT–RCV) use a delay of zero when operating at 9600 bps and higher speeds, and when interfacing with certain mainframes or communications processor with fast turnaround times.

Shared resource processor operating systems differ from CTOS workstation operating systems in the way that they prioritize their internal processes. Because of these differences, any delays configured for BMULTI on a shared resource processor may actually take somewhat longer than specified. The result is that BMULTI installations on the shared resoruce processor may require an increase in the host transmit delay. Experiment with the delays as necessary.

RTS Hold Delay (default = 0)

The number of milliseconds that BMULTI keeps Request–to–Send (RTS) on after the end of a transmission. This delay is used with some older modems or to have the host system modem keep Data Carrier Detect on long enough to ensure that the host receives the transmission.

Anything Downstream? (default = N)

Defines whether BMULTI is operating in a concatenated string of terminals, with other terminals downstream from this one.

Enter *Y* (yes) if the workstation running BMULTI is in the midst of a concatenated string of terminals communicating through a single modem. BMULTI does not reply to a group poll when secondary receive data is on.

Enter *N* (no) if the workstation is the only terminal connected to the modem, if it is the last terminal on a concatenation string, or if it is using TDI. BMULTI ignores secondary receive data when determining its response to a group poll. This parameter does not apply to shared resource processors, which always act as if the value is set to *N*.

TDI on IDM? (default = N)

Defines whether BMULTI is operating through TDI on an IDS or enhanced IDS module. This parameter should be set to *Y* (yes) if the workstation is connected to a data communications network through the TDI interface provided on the IDS module.

Always set this parameter to *N* (no) if the workstation is connected to a data communications network through an RS-232 interface. For example, set this parameter to *N* if using the Data Communications Adapter (DCA).

Number of Stations (default = 16)

The maximum number of addresses on this copy of BMULTI, a number between 1 and 64.

Buffer Size (default = 2048)

The size, in bytes, of the BMULTI Transmit/Receive buffer. The size specified, between 16 and 4096 bytes, determines the size of the receive buffer and the extended message transmit buffer.

ServiceID (default = none)

The ServiceID of this copy of the BMULTI system service, when running under the Multiple Gateways Server (MGS). This field must be left blank or filled in with Service ID of up to 12 alphanumeric characters (no spaces), enclosed by angle brackets (<>). Each copy of BMULTI under MGS must have a unique Service ID and its own configuration file. The system administrator should keep track of all Service IDs assigned.

When BMULTI is installed as a system service under MGS, the Service ID is assigned as the partition name.

DMA? This field enables or disables Direct Memory Access (DMA) functionality for data communications at the communications channel; select *Y* (yes) to enable, *N* (no) to disable.

DMA support is available on the IDS and enhanced IDS modules and on the XE-530 GP/CI board. The DMA hardware is available on channel B on IDS and channels A through H on the XE-530 GP/CI board.

The DMA field can be dynamically reconfigured only if it is defined to be *Y* on system service installation; this is because the buffers required for DMA are only allocated when the system service is installed.

Using DMA Configuration

BMULTI supports the Direct Memory Access (DMA) hardware available on the IDS and enhanced IDS modules and on the XE-530 GP/CI board. DMA should be activated for X.21 connections, but is optional for RS232C (synchronous and asynchronous) and V.35 (synchronous only). With DMA activated, BMULTI supports bit rates up to 64 Kbps, depending on your hardware configuration. DMA should not be used for RS232C or V.35 for speeds less than 9600 bps.

Note: *The RS232C standard does not support speeds greater than 9600 bps. However, reliable communications are generally possible at higher speeds when connected to a DCA/TDI converter through a short cable (three feet long or less).*

If you attempt to install or reconfigure DMA operation for a channel that does not support DMA, the error code 31600, *DMA I/O is not supported by the hardware on the specified channel*, is returned.

BMULTI cannot be dynamically reconfigured for DMA unless it was originally installed with a configuration that uses DMA. These configurations include selecting X.21 leased or switched operation or setting the DMA parameter to *Y*. If an invalid reconfiguration is attempted, the error code 56255 *,DMA initialization attempted after BMULTI successfully installed without DMA*, is displayed.

For best results, the **Bit Rate** parameter in the configuration file should be set to the rate of your communications line. If BMULTI is configured for DMA and transmits more than seven SYNC characters (in synchronous mode) after each transmission (as observed on a data communications monitor, not the BMULTI Line Monitor), you must reset the **Bit Rate** parameter to a higher setting.

Modem Configuration

BMULTI provides auto–dial, auto–answer, and host call–back capabilities through the CTOS Modem Service. Through Modem Service, BMULTI can communicate with Unisys systems with the BMULTI poll/select protocol over synchronous and asynchronous dial–up phone lines. Once a connection is established, all applications using the same copy of the BMULTI system service communicate with the same host. To install multiple copies of BMULTI, use MGS and a unique channel for each BMULTI.

Note: *BMULTI must be installed on the same processor as Modem Service.*

When BMULTI registers as the user of a Modem Service port, no other Modem Service clients can use the port. The maximum bit rate supported is the maximum of the Unisys–supported modems. Refer to the *CTOS Modem Service Operations Guide* for more information.

Figure 4-2 shows the menu displayed when the Modem key (F8) is pressed.

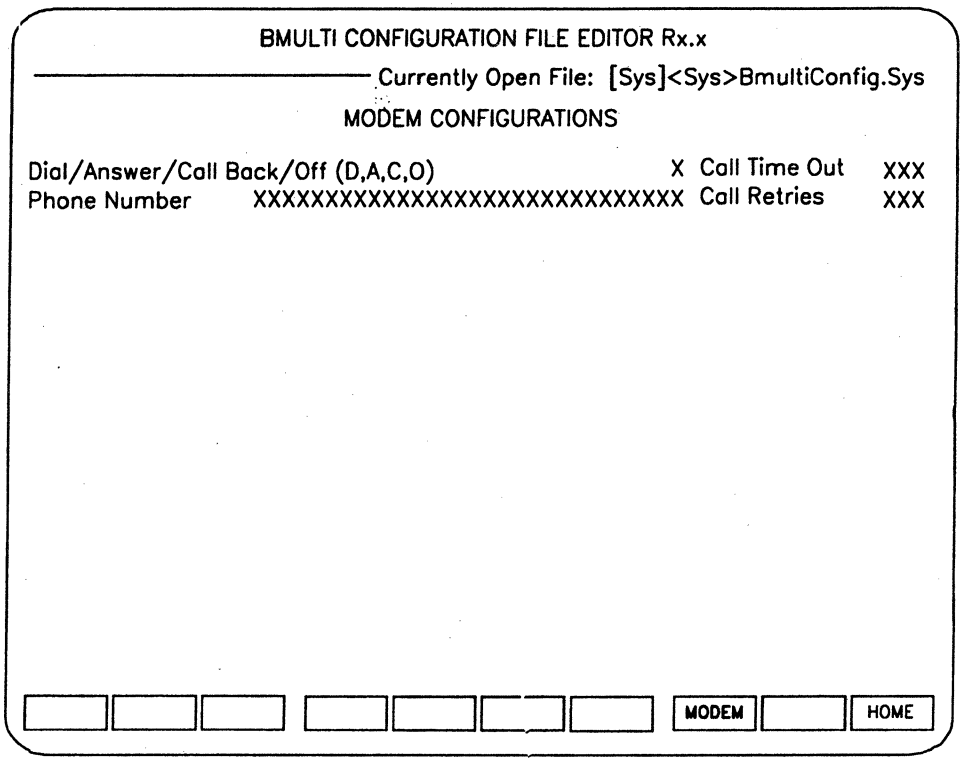

Figure 4–2. BMULTI Modem Configuration Menu

Modem Configuration Menu Parameters

Dial/Answer/ Call Back/Off (D,A,C,O) (default = O)

Specifies whether BMULTI will make outgoing calls, wait to receive calls, or call the host and then hang up so the host can call back. The last option, *O* (Off) disables modem dial–up line functions.

When *D* (Dial) is selected, BMULTI can make outgoing calls. The call is made when the first client application registers with BMULTI. When this option is selected, a phone number must be specified in the **Phone Number** field.

When *A* (Answer) is selected, BMULTI accepts incoming calls. All BMULTI applications must wait until the call is established before registering with BMULTI.

When *C* (Call Back) is selected, BMULTI calls the host, and then hangs up to let the host call back. This scheme is sometimes used for host security purposes. To include a password, the required delay and password characters should be appended to the dialing string (in the **Phone Number** field).

For example, the phone number string could be *555-1234!!!!!999999!!#*; this theoretical string signals the modem to dial 555–1234, wait for 25 seconds for the system to connect (5 seconds for each exclamation point character), transmit the six–digit password 999999, wait ten more seconds, then transmit the pound character (#). The exact characters and modem commands vary for different modems. It will also vary depending on your host modem software. Consult your modem operations manual for more information.

The *O* (Off) option disables Modem Service capabilities.

Dynamic reconfiguration of this parameter from Dial to Answer or vice versa (or to any other setting) must be performed only when no clients are registered with BMULTI.

Call Time Out (default = 60)

Defines the amount of time in seconds the Modem Service waits for a connection when attempting to make a call. This number can vary from 0 to 999. Specifying zero instructs the Modem Service to wait indefinitely. The default value is 60 seconds.

Phone Number (default = none)

Defines the numbers and characters sent to the modem as part of the dialing procecure. The field for this parameter permits a maximum of 30 characters to be used.

All alphanumeric characters are sent to the modem, although the dash, open and close parentheses, and

the space character are ignored. The equals sign instructs the modem to wait for a second dial–tone as part of the dial–up sequence.

This field must contain a valid entry for accessing a host system if the **Dial/Answer/Call Back/Off** field is set to either *D* (Dial) or *C* (Call Back).

To avoid errors when dynamically reconfiguring this parameter, BMULTI should be in the Idle state and not fast–ready, or in the Offline state. If the phone number is changed and the RECNFG key (F5) is pressed while in the main menu, BMULT immediately disconnects the current call, if it is connected, and redials using the new phone number. However, if there are no currently registered users, BMULTI does not make the call using the new phone number until a client registers.

Call Retries (default = 0)

Defines the number of times that the Modem Service retries an outgoing call if there is a failure to connect when dialing. Valid entries range from 0 to 999, where 999 instructs the Modem Service to retry forever. The default value is zero retries.

BMULTI Bit Rate with Modem Service

The BMULTI **Bit Rate** and **Sync or Async** fields are not used by the Modem Service to initialize the Serial Communications Chip (SCC), although these parameters should be configured to the bit rate and mode of the line from the modem to the host. This is because BMULTI's bit rate is used for internal timers and the mode determines whether synchronous or asynchronous communications are to be used.

The Modem Service has its own configuration file in which the bit rate and communications mode are defined. These parameters are located in the Modem Service configuration file, [Sys]<Sys>ModemService.Config.

The Direct Memory Access (DMA) option may be used with Modem Service only if the bit rate is 9600 bps or higher. In addition, DMA communications can only be used on channel B of IDS and enhanced IDS modules and on channels A through H of the XE-530 GP/CI board.

When using DMA, you must make sure that BMULTI's bit rate is the same as the actual bit rate between BMULTI and the host system because BMULTI uses the value for DMA calculations. If your modem is capable of down–shifting to lower speeds, configure the BMULTI bit rate to the highest value used.

Modem Service Channel Selection

When BMULTI is installed as a Modem Service client, BMULTI registers as a user of a specific Modem Service channel or port. Therefore, no other Modem Service clients are permitted to use that channel.

To use more than one channel, you must install multiple copies of BMULTI (under MGS), using the same Modem Service. Each copy of BMULTI must be configured to use a different channel. If you try to install BMULTI using a channel already used by another Modem Service client, the installation will fail and the error condition reported.

The modem used must be defined in the Modem Service configuration file, *[Sys]<Sys>Sys.Modems*. In addition, the parameter *:DCI:* should be set to "Y," regardless of whether the connection is to be synchronous or asynchronous.

Note: *When using asynchronous communications with Modem Service 5.0, BMULTI supports only modems that have a programmable S25 register. The value of the S25 register should be set to 50 or higher. This modem limitation is planned to be eliminated in future releases of Modem Service.*

X.21 Configuration

BMULTI can provide communications over an X.21 leased or circuit–switched lines. To use the X.21 capabilities, you must use port B on the IDS or enhanced IDS module, or port G or H on the XE-530 GP/CI board. In addition, on the XE-530, a connector converter is required to change from the 25–pin D–connector to a 15–pin D–connector. To operate over circuit–switched lines, the BTOS X.21 Circuit Switching Service (CSS). BTOS X.21 CSS is not required for leased line operation.

DMA (Direct Memory Access) should be activated for X.21 connections and supports bit rates up to 64 Kbps (depending on your hardware configuration). BMULTI does not use X.21 byte timing in any X.21 mode of operation.

X.21 Leased Line Functionality

In an X.21 leased line environment, BMULTI transfers data in one of two modes, normal mode and control switching mode.

Under the normal mode of operation, BMULTI remains in the Ready state (state 1) until it detects that the Indicator line, as set by the host, is in the ON condition. When this happens, BMULTI sets the Control line to ON, entering the Data Transfer state (state 13; for information about BMULTI states, refer to Section 8). BMULTI remains in the Data Transfer state until it detects that the Indicator line has changed to the OFF condition. At that time, BMULTI sets the Control line to OFF, entering the Ready state again. In the normal mode, BMULTI and the host can both transmit at the same time (full duplex).

On the enhanced IDS module and on the XE-530 GP/CI board, another mode of operation is available, in addition to normal mode On these systems, BMULTI can enter a mode where data is transferred in states 13S and 13R only. A configurable option, **Control Switching**, enables this mode. In this mode, both BMULTI and the host remain in the Ready state (state 1) until one of them has data to transmit. When this happens, the transmitting party turns their Control line to ON (entering state 13S) and begins transmitting data. When transmission is finished, the transmitter sets the Control line back to OFF. If the Control line goes ON while BMULTI is transmitting (entering the Data Transfer state, state 13), BMULTI returns to the Receive state (state 13R) and receives the host data. In the control switching mode, BMULTI and the host take turns transmitting (essentially half–duplex operation).

X.21 Circuit Switching Functionality

BMULTI uses the BTOS X.21 Circuit Switching Service (CSS) to enable communications over X.21 switched lines. BMULTI supports the following features:

- Call direction can be incoming, outgoing, or both (default is both)
- Up to 64 different host addresses can be configured (one per BMULTI client application)
- Configurable timer options allow calls to be cleared when the line is idle
- Closed User Group (CUG) index is configurable for each host address that is called

- Reverse charging is configurable for each host address called
- Full, abbreviated, and direct call types can be selected for outgoing calls
- BMULTI can be configured to instruct the BTOS X.21 CSS to perform incoming call address verification; this feature may be used to screen out unwanted calls

BMULTI operates in a Contention mode when communicating over X.21 switched lines, and interprets clearing of the call as a Go–to–Contention sequence; this is true even if BMULTI itself cleared the call. If the host calls, BMULTI responds to polls and selects. If a BMULTI application has data to transmit to the host (and outgoing calls are enabled), BMULTI makes the call to the host and sends a Wake–Up–From–Contention sequence for that address.

BMULTI expects to run in a poll/select contention mode when operating over switched lines. When the call is dropped, either by the host or by BMULTI, the host should behave as if it has just sent the Go–to–Contention sequence. If BMULTI makes the next call to the host, it sends a Wake–Up–From–Contention sequence for the station requesting to transmit.

X.21 itself is a circuit–switching protocol (unlike X.25, which is packet switching). A call is established whenever there is data to be transfered. The call establishment time is very fast, and you are usually charged for the connection time, not the amount of data transfered. However, some X.21 PDNs charge for a fixed initial set up time. You can use the X.21 **Call Hold Time** parameter to keep the call connected for a fixed period of time after the call is connected.

X.21 Circuit–Switched Host Identification

For incoming calls, BMULTI needs to determine which host is calling so that it can determine which application stations to activate. Once the incoming call address verification (if selected) has passed, BMULTI attempts to identify the calling host as follows:

- First, if the Circuit Switching Network provides the calling number as "DCE Provided Information," BMULTI attempts to find a match between the number and the X.21 host addresses as they are configured in the BMULTI configuration file.

- If the DCE provided number does not match any of the configured X.21 host addresses, BMULTI uses the poll/select addresses from the incoming polls and selects to determing with which host it is communicating. It continues to compare the incoming poll/select addresses with its own list of poll/select addresses from the configuration file. When a match is found, the configured Host Index parameter identifies the host for the current call.

When determining the index of the calling host, the DCE provided number is always a full address specification. Therefore, DCE provided addresses do not always match up with configured abbreviated or direct addresses.

If two hosts have the same poll/select address configured, BMULTI may mistake one host for the other and may respond in error. In addition, BMULTI has only one group poll address for all hosts. If a host calls and BMULTI cannot identify the calling host, BMULTI does not respond to the group poll.

Suggestions for improving host identification:

- If possible, you should subscribe to a Circuit Switching Network service that provides the address of the calling party. With this service, the BTOS X.21 CSS can perform incoming call address verification and BMULTI can identify the calling host by its address. If this facility is subscribed to, use full addressing for all configured X.21 host addresses.
- If the calling address cannot be provided, or if abbreviated or direct addressing is necessary, make sure that all hosts calling BMULTI use unique poll/select addresses and that they do not send group polls.

X.21 Configuration Menu

Figure 4-3 shows the menu displayed when the X21Parm key (F9) is pressed.

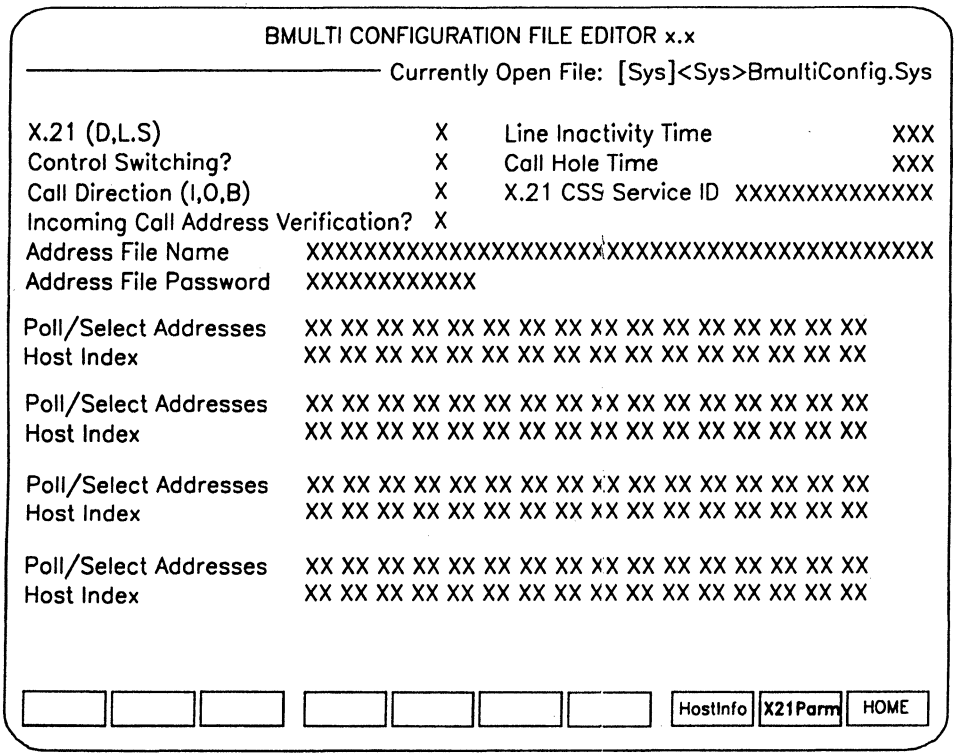

Figure 4–3. BMULTI X.21 Configuration Menu

X.21 Configuration Menu Parameters

X.21 (D,L,S) (default = D)

Selects the type of X.21 operation. In this field, the default, *D*, disables X.21 communications. *L* selects leased line operation, and *S* selects circuit–switched operation. The default is *D* (disabled).

When leased line operation is selected, the only other parameter enabled is the **Control Switching?** parameter. When using a leased line, X.21 CSS is not needed; however, the communications channel must support X.21 communications (channel B on the IDS and enhanced IDS modules and channel H on the XE-530 GP/CI board).

When circuit switching is selected, all X.21 parameters listed below *except* for **Control Switching?** are active.

Control Switching? (default = N)

Enable or disable Control Switching. Control switching is available *only* on the enhanced IDS module and the XE-530. This parameter can be set to *Y* (yes) to enable the Control Switching mode for data transfer in leased line operation. The setting *N* (no), the default, disables the mode.

The setting of this parameter is irrelevant if the **X.21 (D,L,S)** parameter is set to disable X.21 operation or to X.21 circuit–switched operation.

All of the following parameters are for X.21 circuit–switched operation:

Line Inactivity Time (default = 0)

Determines how long BMULTI allows a line to be inactive before clearing the call. The timer is started initially when the call is connected and restarted whenever data is transmitted or received.

This timer can be set from any value from 0 to 65535 tenths of a second. The value zero (the default) disables this timer.

Call Hold Time (default = 0)	Determines how long BMULTI allows any call to be connected, regardless of line activity. The timer is started when the call is connected. This timer can be set from any value from 0 to 65535 tenths of a second. The value zero (the default) disables this timer.
Call Direction (I,O,B) (default = B)	Determines whether BMULTI accepts incoming calls (*I*), makes outgoing calls (*O*), or both (*B*). The default is *B*, both.
X.21 CSS Service ID (default = none)	Defines the Service ID for the X.21 CSS. If multiple copies of the X.21 CSS are running under the Multiple Gateways Server (MGS), enter the Service ID of the desired X.21 CSS in this parameter field. The Service ID must be no longer than 12 characters and enclosed in angle brackets.
Incoming Call Address Verification (default = N)	Determines whether BMULTI performs incoming call address verification. Set this parameter to *Y* (yes) to enable verification or to *N* (no) to disable it. The default is *N*. If this parameter is set to *Y*, you must enter the name of a valid address file in the **Address File Name** field.
Address File Name (default = none)	Defines the file name of a valid address file for the purpose of incoming call address verification. Refer to the *BTOS X.21 Circuit Switching Service (CSS) Operations Guide* for information on the proper format of this file. The default for this parameter is no file name.
Address File Password (default = none)	Contains the (optional) password for the address file, as indicated by the **Address File Name** parameter. The default is no password.
Poll/Select Addresses (default = none)	Specifies all client poll/select addresses, using unique addresses for each. Each address can be any two alphanumeric characters, and each address specified acts as a virtual address; that is, BMULTI responds to polls and selects for those addresses, even if they are not currently registered with BMULTI.

The default for these fields is no address. Therefore, for BMULTI to function and communicate with a host using an X.21 connection, at least one address must be entered. Up to 64 addresses can be defined.

Host Index (default = none)

Defines the host index entries for every configured poll/select address. This entry is a number from 1 to 64, corresponding to the Host Index number in the Host Information configuration menu.

The Host Information configuration menu defines the X.21 host address, whether reverse charging is to be used, and the Closed User Group (CUG) index. To activate the Host Information configuration menu, press the HostInfo key (F8) while in the X.21 configuration menu. The first screen defines host index entries numbered 1 to 32, the second screen for entries 33 to 64; press the HostInfo key (F8) to toggle between these two screens.

X.21 Host Information Configuration Menu

When in the X.21 configuration menu, an additional option is displayed for function key F8, HostInfo. Pressing this key activates the X.21 Host Information configuration menu. Figure 4-4 shows the first menu displayed when the HostInfo key is pressed. In it, host index entries 1 through 32 are defined; a second screen defines host index entries 33 through 64. Pressing the HostInfo key repeatedly toggles between the two screens. You can press the X21Parm key to return to the X.21 parameter menu or the HOME key to return to the main configuration menu.

BMULTI CONFIGURATION FILE EDITOR Rx.x

Currently Open File: [Sys]<Sys>BmultiConfig.Sys

HOST INFORMATION

Host Index	X.21 Address	Rev Chg?	CUG Indx	Host Index	X.21 Address	Rev Chg?	CUG Indx
1	xxxxxxxxxxxxxx	x	xx	2	xxxxxxxxxxxxxx	x	xx
3	xxxxxxxxxxxxxx	x	xx	4	xxxxxxxxxxxxxx	x	xx
5	xxxxxxxxxxxxxx	x	xx	6	xxxxxxxxxxxxxx	x	xx
7	xxxxxxxxxxxxxx	x	xx	8	xxxxxxxxxxxxxx	x	xx
9	xxxxxxxxxxxxxx	x	xx	10	xxxxxxxxxxxxxx	x	xx
11	xxxxxxxxxxxxxx	x	xx	12	xxxxxxxxxxxxxx	x	xx
13	xxxxxxxxxxxxxx	x	xx	14	xxxxxxxxxxxxxx	x	xx
15	xxxxxxxxxxxxxx	x	xx	16	xxxxxxxxxxxxxx	x	xx
17	xxxxxxxxxxxxxx	x	xx	18	xxxxxxxxxxxxxx	x	xx
19	xxxxxxxxxxxxxx	x	xx	20	xxxxxxxxxxxxxx	x	xx
21	xxxxxxxxxxxxxx	x	xx	22	xxxxxxxxxxxxxx	x	xx
23	xxxxxxxxxxxxxx	x	xx	24	xxxxxxxxxxxxxx	x	xx
25	xxxxxxxxxxxxxx	x	xx	26	xxxxxxxxxxxxxx	x	xx
27	xxxxxxxxxxxxxx	x	xx	28	xxxxxxxxxxxxxx	x	xx
29	xxxxxxxxxxxxxx	x	xx	30	xxxxxxxxxxxxxx	x	xx
31	xxxxxxxxxxxxxx	x	xx	32	xxxxxxxxxxxxxx	x	xx

HostInfo | X21Parm | HOME

Figure 4–4. BMULTI X.21 Host Information Configuration Menu

X.21 Host Information Configuration Menu Parameters

X.21 Address (default = none)

Defines a unique X.21 network address of up to 14 digits for use on the Circuit Switching Network. This address is supplied to you when if you subscribe to a network.

A direct address is specified by a single dot (.)

An abbreviated address is defined by a dot (.) followed by some digits (up to 13).

A full address has no dot prefix.

Rev Chg? (default = N)

Selects whether the network will charge the host for all calls that BMULTI makes to that host. Select *Y* (yes) to enable reverse charging, or *N* (no) to disable it *N* is the default.

To use reverse charging, you must subscribe for this option with the supplier of the Circuit Switching Network.

CUG Index (default = none)

Specifies the Closed User Group (CUG) index to use when making a call to the host. The host and/or Circuit Switching Network may or may not use this information.

X.21 Configuration Errors

If you attempt to install or reconfigure BMULTI for X.21 operation on a port that does not support X.21, an error is reported.

In X.21 circuit–switched operation, if no X.21 poll/select addresses are configured, BMULTI reports error 32796, *Entry Error*; if duplicate addresses are configured, BMULTI reports error 32794, *Duplicate virtua address or Duplicate X.21 poll/select address.*

During dynamic reconfiguration, if a client application is already active with BMULTI and using a specific poll/select address, and you remove that address from the new list of poll/select addresses, BMULTI reports error 56254, *Active poll/select address not specified.*

If you attempt to install or reconfigure BMULTI for X.21 circuit–switched operation and the X.21 Circuit Switching Service (CSS) is not available, error 33, *Service not available*, is reported.

If other errors occur after installation, such as the network going into the Not Ready state or communications being disconnected, BMULTI will continue to try to open the X.21 port. These attempts can be observed through the X.21 CSS Status Monitor (not the BMULTI status monitor). In addition, applications equipped with connection status reporting (for example, BTE 3.1 or higher) will be notified if the state of a station connection changes.

Installing BMULTI

Installing BMULTI with MGS

If installing BMULTI under Multiple Gateways Server (MGS), MGS must be installed as a system service *before* installing BMULTI.

In addition, the BMULTI configuration file(s) must include a valid (and unique) Service ID in the Service ID field.

Manual Installation on a CTOS Workstation

Once edit the BMULTI configuration file or files, using the BMULTI Configuration File Editor, you can install BMULTI, that is, load it into RAM as a system service.

1. Type *Install BMULTI Gateway* and press the RETURN key.
2. Enter the information in the form that appears

   ```
   Install BMULTI Gateway
     [BMULTI Configuration file]
   ```

   If you do not enter a file name in the **[BMULTI Configuration file]** field, it will use the default file name [Sys]<Sys>BmultiConfig.Sys.
3. When you are finished entering the information, press the GO key.

Manual Installation on the IDS Module

As an alternative to loading BMULTI on the CPU of a CTOS workstation, you can instead load it on an IDS or enhanced IDS module attached to the workstation. When running on the IDS module, the CPU processor overhead for BMULTI is considerably lower than when running on the CPU module.

Note: *To install BMULTI on the IDS module, you must have the module attached to the CTOS workstation, and IDMSS software installed on it.*

Once you edit the BMULTI configuration file (or files), you can install BMULTI on the IDS module, that is, load it onto module as a system service.

1. Type *IMRun* and press the RETURN key.
2. Enter the information in the form that appears

```
IMRun
  Processor ID  #
  Run File      [Sys]<Sys>BmZip.Run
  [Case]
  [Command]
  [Parameter 1] ConfigurationFileName
  [Parameter 2]
 (etc...)
```

   The value # is the processor ID number of the IDS module, and *[Sys]<Sys>BmZip.Run* is the name of the BMULTI system service loader file.

   If you do not enter a file name in the first optional parameter field, BMULTI will use the default file *[Sys]<Sys>BmultiConfig.Sys*.

3. When you are finished entering the information, press the GO key.

Automatic Installation on a CTOS Workstation at Bootup

To install BMULTI on a CTOS workstation so that it is available at bootup, edit the file *[Sys]<Sys>SysInit.Jcl*. Enter the line:

`$RUN [Sys]<Sys>BmZip.Run,` *ConfigurationFile*

(where *ConfigurationFile* is the appropriate value)

If you do not enter a value for *ConfigurationFile*, BMULTI uses the default name [Sys]<Sys>BmultiConfig.Sys.

Automatic Installation on the IDS Module at Bootup

To install BMULTI on an IDS module attached to a CTOS workstation so that the BMULTI gateway is available at bootup, edit the file *[Sys]<Sys>SysInit.Jcl*. Enter the line:

```
$IMRUN #,[Sys]<Sys>BmZip.Run,,,,ConfigurationFile
```

(where # is the processor ID number, and *ConfigurationFile* is the appropriate value)

If you do not enter a value for *ConfigurationFile*, BMULTI uses the default name [Sys]<Sys>BmultiConfig.Sys.

Automatic Installation on the Shared Resource Processor at Bootup

Note: *There is no manual method of installing BMULTI as a system service on a shared resource processor.*

To install BMULTI on a shared resource processor so that it is available at bootup, edit the file system initialization JCL batch file.

On the XE520, this file is either [Sys]<Sys>InitCP*xx*.Jcl or [Sys]<Sys>InitTP*xx*.Jcl for cluster and terminal processor boards, respectively. The number *xx* represents the board number on your system.

On the XE-530, this file is called [Sys]<Sys>SRPSysInit.Jcl. This file is divided into sections, one for each of the cluster or terminal processor boards used.

In the appropriate JCL file, enter the line:

```
$RUN [Sys]<Sys>BmZip.Run,ConfigurationFile
```

(where *ConfigurationFile* is the appropriate value)

If you do not enter a value for *ConfigurationFile*, BMULTI uses the default name [Sys]<Sys>BmultiConfig.Sys.

Deinstalling BMULTI

The **Deinstall BMULTI Gateway** command removes BMULTI from background processing. This allows you to change installation parameters and reinstall the system service without having to reboot.

- When loaded on a CTOS workstation, you must deinstall BMULTI on the workstation on which you installed it. You cannot deinstall BMULTI from a remote node.
- You must finish any application that uses BMULTI before using the deinstall utility.
- The deinstall utility must be executed in the primary partition of the workstation in which BMULTI is running. You cannot, therefore, deinstall BMULTI when running Context Manager.
- You cannot deinstall BMULTI from a shared resource processor or th IDS or enhanced IDS module.

To execute the deinstall utility from the Executive, type *Deinstall BMULTI Gateway* on the Executive command line and press the RETURN key. Enter the information in the form that appears:

```
Deinstall BMULTI Gateway
  [<ServiceID>]
```

If BMULTI was installed under MGS, enter the Service ID for the copy o BMULTI you want to deinstall. Press the GO key.

Section 5
The Status Monitor

Use the status monitor to determine which applications are using BMULTI and how busy they are. You can also use the status monitor to purge an address, unless you are running the monitor from a remote node.

Entering the Status Monitor

To activate the status monitor, type *BMULTI Status* and press the RETURN key. The following is displayed:

```
BMULTI Status
  [{Nodename}<ServiceID>]
```

You can monitor BMULTI running on a remote BNet node or one of possibly several copies of BMULTI running on the same system under the BTOS Multiple Gateways Service (MGS).

- If applicable, enter the node name and Service ID. The node name, if entered, must be enclosed in braces; the Service ID is enclosed in angle brackets.
- If there is a node name, but no Service ID, enter only the node name. Similarly, if there is a Service ID, but you want to monitor BMULTI on the local system, enter only the Service ID.
- If there is no node name or Service ID, leave this field blank.

Once the field is filled in (if applicable), press the GO key to enter the status monitor.

Figure 5-1 illustrates a sample status monitor display. The display is updated at one–second intervals. Short messages transmitted from a client station may be completed before the display is updated, causing it to appear as if no transmission occurred.

The line displayed in dim reverse video is called the selected address.

BMULTI STATUS MONITOR 10.1

VERSION: R10.1.0000 INSTALLED: At Master ON CHANNEL: B

STATIONS IN USE : 16 NAK COUNT: 201 LINE : Active

#	ADDR	PROGRAM	USER	STATE	WS	RQs	TIME ON	SENT	RECD
01	1a	MT983 Em	Admin	Rcv Rdy	32	1	180	101	100
02	1b	FT2	Renee	Xmt Rdy	03	1	2880	45	45
03	1c	DTS	Accntng	Idle	21	0	120	75	75
04	1d		User	Rcv Rdy	41	1	60	20	120
05	1e	InfoLink		Xmting	17	2	7200	95	95
06	1f	PPT		Rcv Rdy	31	1	1440	1	50
07	1g	T 27 Em	Donald	Xmt/Rcv	51	2	240	201	200
08	1h		Maggie	Idle	62	1	5	10	10
09	01	DOCxfer	WRITE2	Rcv Rdy	32	1	30	60	60
10	02	BTE	Robert	Xmt Rdy	03	1	480	55	45
11	03		Grendel	Idle	21	0	77	12	34
12	71	Infoview	Amy	Rcv Rdy	41	1	180	5	1
13	72	Infoview	Sftwre	Xmting	17	2	180	4	2
14	73	Infoview	Ursa	Rcv Rdy	31	1	180	3	3
15	74	Infoview	Dvlmpt	Xmt/Rcv	51	2	180	2	4
16	75	Infoview	Supprt	Idle	62	1	180	1	5

Figure 5–1. Sample BMULTI Status Monitor Display

Status Monitor Display Fields

Version	Lists the BMULTI release level.
Installed	Shows type of workstation on which BMULTI is installed (*At Master*, *Local*, or *Remote*), or the Service ID.
On Channel	Indicates the channel on which BMULTI is running.
# Stations in Use	Shows the number of workstations using BMULTI.
NAK Count	Indicates the number of NAKs BMULTI sends in reply to messages. This helps you determine line quality. (Not counted are NAKs sent by BMULTI in response to select requests or NAKs received by BMULTI.)
Line	Active or inactive. This indicates whether the line was used during the previous second.
Addr	Indicates the BMULTI address, a hexadecimal number.
Program	The application, such as BTE or File Transfer, running at a particular address.
User	The signon identification of the user at that address.
State	Describes the BMULTI state. (See Section 8, "BMULTI State Machine," for more information.)
Ws	Shows the workstation number, assigned by the operating system.
Rq	Shows the number of requests outstanding.
Time on	Shows the number of minutes since the application was initiated.

Sent	Shows the number of messages transmitted since application was initiated.
Rcd	Number of messages received since application was initiated.

Displaying Additional Addresses

If there are more addresses online than your screen can show, use the scrolling or paging keys to display the additional addresses.

How to Purge an Address

In the rare event that you need to purge (free) an address, you can easily do so while you are running the status monitor, unless you are accessing a remote BNet node:

1. Type *BMULTI Status* and press the RETURN key; remember to enter the node name and/or Service ID, if applicable. Press the GO key.
2. The address line displayed in dim reverse video is called the selected address. Use the arrow keys to select the address you need to purge.
3. Press the MARK key. The selected address line changes to bright reverse video.
4. To purge the selected address, press the DELETE key. If you decide not to purge the marked address, press the CANCEL key.

Exiting the Status Monitor

To exit the status monitor, press the FINISH key.

Section 6
The Line Monitor

The line monitor is an software facility that shows all data passing through BMULTI. For this reason, installing the line monitor during the Software Installation procedure is optional. Because some of line data may be proprietary, you may want to load the monitor from a floppy disk.

When you use the line monitor, data from BMULTI is saved in a ring buffer (a ring buffer is one in which the newest information overwrites the oldest). The ring buffer size can be configured from 1 Kbyte to 60 Kbytes. The default size is 10 Kbytes.

The line monitor looks for the file [Sys]<Sys>BmLineMonitorFont.*xxx* (where *xxx* represents the type of workstation video, and can take the values B25, B27, VE1, or AGP). If the appropriate file is not located, the appearance of line monitor data may be affected, however the data is not changed.

Using the monitor's menu, you can:

- "Freeze" the screen display while data is being received
- Store up to 60 Kbytes of data in a buffer
- Save any portion of buffered data to a disk file
- Scroll through buffered data using the page and scroll keys, or the "jump" function
- Load and examine data that was previously saved by the line monitor

Entering the Line Monitor

To enter the monitor from the Executive, type *Monitor BMULTI* and press the RETURN key. The screen displays:

```
Monitor BMULTI
  [<ServiceID>]
```

If applicable, enter the Service ID for the specific copy of BMULTI running under the Multiple Gateways Server (MGS). The Service ID, if entered, is enclosed in angle brackets.

When in the line monitor, you can access online help by pressing the HELP key. Online help will give you a brief description of commands and types of error messages.

Figure 6–1 shows the initial line monitor display.

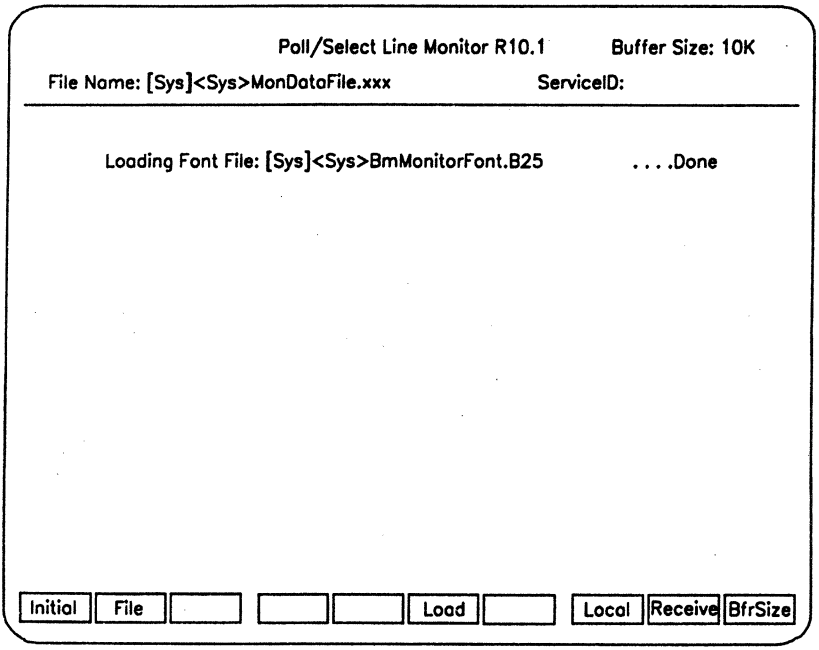

Figure 6–1. Initial Line Monitor Display

Function Key Menu and Line Monitor States

The line monitor has two states, Local (offline) and Receive (online). In the Local state, the line monitor is logged off from BMULTI and under operator control. In Receive state, the line monitor is logged onto BMULTI. There are two function key menus in the local state: Initial (Figure 6-2) and Local (Figure 6-3). There is only one function key menu in Receive state (Figure 6-4).

When you first enter the line monitor, the Initial menu is displayed. As soon as you perform any line monitor function, either the Local or the Receive menu is displayed. A function key label is always highlighted; this indicates which menu is being displayed or which option was chosen from the menu.

Figure 6–2. Initial Function Key Menu

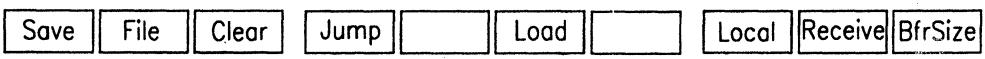

Figure 6–3. Local Function Key Menu

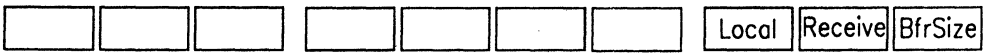

Figure 6–4. Receive Function Key Menu

To perform functions such as *Save* and *Jump*, you must be logged off from BMULTI, in the local state. The function key label *Local* is highlighted. If Receive or Freeze is highlighted, you are logged on to BMULTI. Press the Local key to log off from BMULTI and return to the Local state.

File Name

The default file name is [Sys]<Sys>MonDataFile.XXX. This file name is automatically given to any data files you receive or save, unless you change the default file name or specify another file name when you load or save a file.

To change the default file name for a particular session:

1. Press the File key (F2). The current default file name is displayed.
2. Type the name of the file that you want. To correct errors, press the BACKSPACE key or the left–arrow key.
3. Press the GO key.

Ring Buffer

Screen Messages

Information about the buffer appears on the top two lines of the screen.

Bytes in Buffer (Bytes in Bfr)	Number of bytes currently in the buffer.
Buffer Size	The number of bytes allocated for the ring buffer.
Buffer Position (Buffer Posn)	The offset of the first byte of data visible in the display window. This number is zero relative, meaning that the first offset in the buffer is numbered 0, the second, 1, and so on.

Changing the Buffer Size

Note: Changing the buffer size clears the buffer.

Buffer size can range from 1 to 60 K. To set the buffer size:

1. In Local state, press the BfrSize key (F10). The screen clears and displays the current buffer size.
2. Type the number of kilobytes to be allocated to the buffer and press the GO key.

Viewing Data from BMULTI

To monitor BMULTI, press the Receive key (F9). The line monitor is then logged onto BMULTI. As the screen fills with data, the newest data overwrites the oldest; data transmitted to the host appears in reverse video. A line is cleared ahead of the new data to indicate where new data begins. Data is sent to a ring buffer as described previously. Figure 6-5 illustrates a line monitor screen receiving test pattern "abcde."

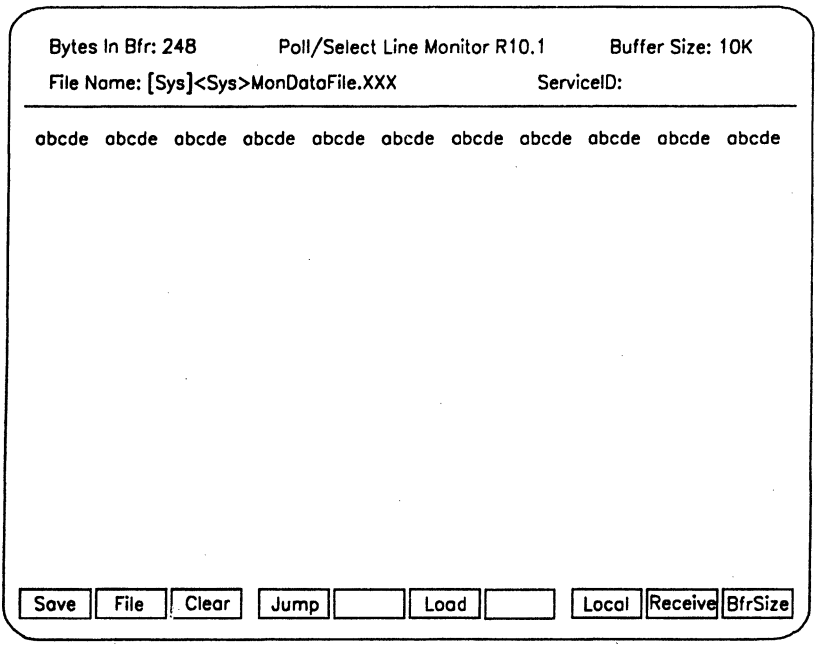

Figure 6–5. Sample Receive Session

Freezing when Receiving Data

You can examine data more closely by pressing the Freeze key (F10) or any unassigned key. This function stops the flow of data to your screen. Freeze is a toggle; press it again to update the screen.

Although the screen is frozen, data continues to be sent to the ring buffer. When the screen is unfrozen, the screen displays the new information that was received while the screen was frozen.

Exiting Receive

To return to the Local or Offline state, press the Local key (F8) or the CANCEL key. The window displays the last data received, although the menu options and captions change. Figure 6-6 shows the screen display from Figure 6-5 after exit from receive.

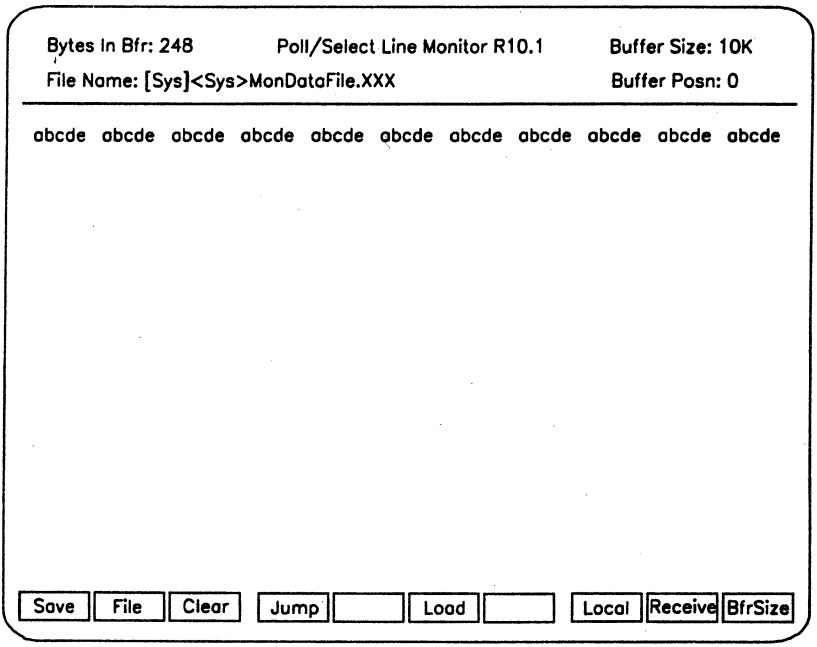

Figure 6–6. Screen after Exit from Receive

Loading a File to the Ring Buffer

You can load a file that was previously saved by the line monitor to the ring buffer. You can then use functions such as Jump to manipulate and/or examine the data.

Press the Load key (F6). Enter the name of the file you want to examine and press the GO key. Your window displays the file you loaded.

Use the Jump key (F4) and the SCROLL UP, NEXT PAGE, PREV PAGE, and SCROLL DOWN keys to manipulate the screen display. Use the Save key (F1) to save small portions of the file for closer study.

Save

All or part of a buffer can be saved using this function. From the local state:

1. With the data visible on the display, note the first offset and the number of bytes you want to save. The offset is the location in the buffer, with one character occupying one space. For example, offset 10 is the tenth character in the buffer.
2. Press the Save key (F1).
3. Enter a file name if you do not want to use the default and press the GO key.
4. The display prompts you for the offset and the number of bytes to be saved. Enter the location of the first offset you want to save.
5. The number of bytes left in the buffer automatically appears. Enter the number of bytes you want to save and press the GO key.

Clearing the Buffer

From the Local state, press the Clear key (F3) to erase the buffer.

Jumping and Scrolling Through Data

To jump to a specific offset in the buffer, use the Jump key (F4) from the Local state. The display begins with the offset you specify. You may then use the scrolling and paging keys to examine the information surrounding the chosen offset.

Figure 6-7 illustrates a file with an error at offset 97. Figure 6-8 illustrates this same file after the operator "jumped" to offset 89.

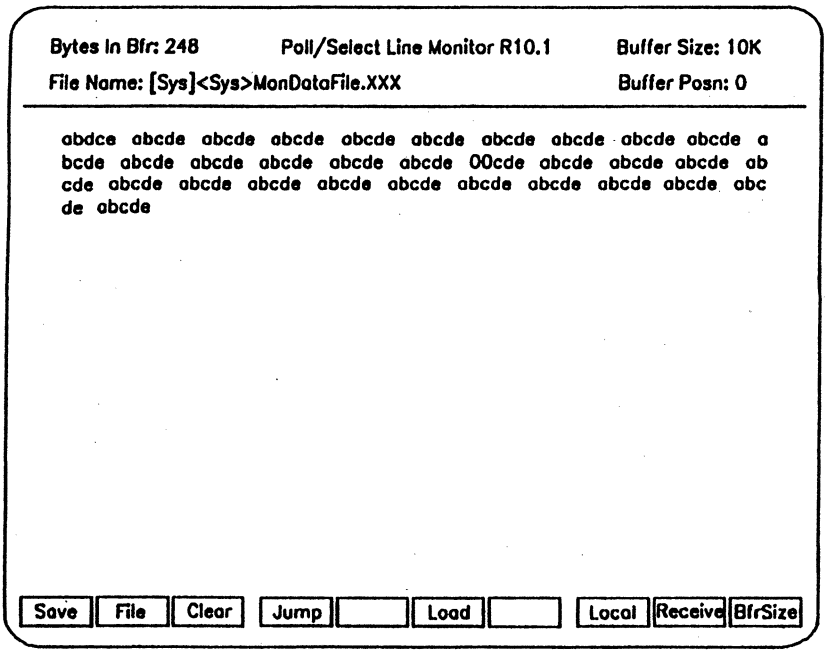

Figure 6–7. Sample Buffer with Error "00cde" at Offset 97

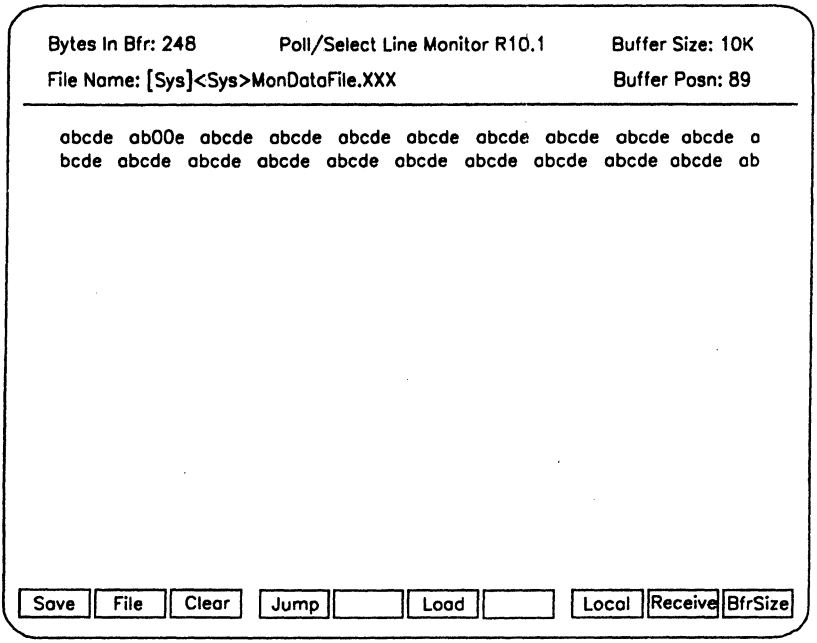

Figure 6–8. Buffer Shown in Figure 6-7 After Jump to Offset 89

To jump:

1. Press the Jump key (F4).
2. Enter the offset to which you want to jump. The offset location is zero relative, meaning that the first offset in the buffer is number 0, the second offset is number 1, and so on.
3. Press the GO key.

Line Monitor Display when DMA is Active

The line monitor displays data in a slightly different fashion when DMA (Direct Memory Access) is used. Received data is displayed as it is processed by BMULTI, in the same way it is displayed when you are not using DMA. Transmitted data, however, appears in blocks. This is particularly noticeable at lower bit rates.

The first character of a transmission (in synchronous transmissions, this is usually the SYNC character) is sent to the line monitor immediately. The remaining characters are not sent to the line monitor until they have been transmitted by the DMA controller. Because BMULTI only checks the DMA controller periodically, the transmitted data is displayed in blocks. If characters are received while a transmission is in progress, all transmitted characters are processed and displayed first on the line monitor screen.

Exiting the Line Monitor

To exit the line monitor, press the FINISH key.

Troubleshooting with the Line Monitor: A Sample Walkthrough

Suppose BMULTI seems to be working abnormally, and you suspect that this may have to do with the data being sent over the communications lines. These steps show one way to use the line monitor as a troubleshooting tool:

1. Enter the line monitor.
2. Change the file name (optional).
3. Set the buffer to an appropriate size.
4. Press the Receive key (F9).
5. Watch the data as it appears, freezing the display as needed. Data sent to the host will appear in reverse video. When you see information that appears to be unusual, return to the Local state.
6. Press the Jump key (F4) and use the scrolling and paging keys to identify a sample of the file you want to save. Note the appropriate offset numbers.
7. Save this subset.
8. Later, you can load the saved file to the line monitor for further examination.

Section 7
Basic BMULTI Concepts

This section contains recommendations for programming BMULTI and explains basic concepts. Sections 8 and 9 ("BMULTI State Machine" and "Protocol Description," respectively) also acquaint you with Unisys poll/select data communications.

To make use of BMULTI, you must have an existing application or you must create one. To write the software, use the procedural interfaces described in Sections 10 and 11 ("High–Level Procedural Interface" and "Low–Level Procedural Interface"), and Appendix G ("Sample Programs").

The procedural interfaces are object modules residing in *Bmulti.Lib*, the BMULTI procedural interface library. They format and issue requests to the BMULTI service. There are two procedural interfaces available:

- The High–Level Interface (HLI) is suitable for use by application programmers
- The enhanced Low–Level Interface (LLI) is designed for experienced programmers who want a high degree of control over the operation of BMULTI; to use the BMULTI Suspend/Resume terminal redirection facilities, you must use the LLI.

Note: *BMULTI no longer supports applications that were written using the Multiple–Task Interface (MTI) and Single–Task Interface (STI) from previous versions of BMULTI. You should use the HLI and LLI interfaces to rewrite existing applications or to create new ones.*

Recommended Interfaces

Programming Task	Language	Interface
New batch application	Pascal, C, COBOL, BASIC	HLI
New interactive application	Pascal, C	LLI
Enhancing older application	Any	HLI/LLI
Using the BMULTI Suspend/Resume filter	Any	LLI

BMULTI includes support for extended character sets. This support is incorporated into BMULTI so that this feature need not be included in each individual application; it is available in one place for all applications running with BMULTI.

Linking Applications with BMULTI.Lib

After writing and compiling your program, you must link it with BMULTI.Lib. If you are using BASIC Interpreter or COBOL applications, link the interpreter with BMULTI.Lib.

Table 7-1 provides information about BMULTI.Lib modules.

Table 7–1. BMULTI.Lib Modules and Procedure Calls

Interface Used	BMULTI.Lib Module	Procedures Served
Multitasking Low–Level Interface	BMX1	BmCommand BmReport BmReportWait BmOpenII BmResume BmSuspend
	BMX2	BmIdentify BmQuery BmStatus
Multitasking High–Level Interface	HLI6	CloseBMULTI OpenBMULTI ReadBMULTI ResetBMULTI SetOptionBMULTI

Note: The object module HLI6 uses the module BMX1.

Language Configuration

Before using FORTRAN, COBOL, or BASIC with BMULTI, you must regenerate the language interpreters and/or run time support modules. Use the Editor to add certain lines to the *.Asm* file associated with your language (either COBOLGen.Asm, BasGen.Asm or ForGen.Asm), assembling it, and either relinking the interpreter or, in the case of FORTRAN and compiled BASIC, linking the resultant object module with the object module that resulted from the compile.

The following lines must be added to the appropriate *.Asm* file at the locations indicated by comments in each file, below the comment add new entries.

BASIC (BasGen.Asm)

```
%TableEntry(1,14,OPENBMULTI)
%TableEntry(1,8,READBMULTI)
%TableEntry(1,8,WRITEBMULTI)
%TableEntry(1,4,SETOPTIONBMULTI)
%TableEntry(1,2,RESETBMULTI)
%TableEntry(1,2,CLOSEBMULTI)
%TableEntry(1,24,BMOPENII)
%TableEntry(1,10,BMCOMMAND)
%TableEntry(1,6,BMREPORT)
%TableEntry(1,6,BMREPORTWAIT)
%TableEntry(1,8,BMREPORTTIMEOUT)
%TableEntry(1,24,BMGETSTATUS)
%TableEntry(1,6,BMQUERYHANDLE)
%TableEntry(1,6,BMSETXLATMODE)
%TableEntry(1,6,SELECTBMULTI)
%TableEntry(1,4,SETXLATMODEBMULTI)
```

COBOL (COBOLGen.Asm)

Add these lines if you are using COBOL release 4.0 or lower:

```
%TableEntry(0,w,OPENBMULTI,5,w,b,w,r,r)
%TableEntry(0,w,READBMULTI,3,b,r,w)
%TableEntry(0,w,WRITEBMULTI,3,b,r,w)
%TableEntry(0,w,SETOPTIONBMULTI,2,b,b)
%TableEntry(0,w,RESETBMULTI,1,b)
%TableEntry(0,w,CLOSEBMULTI,1,b)
%TableEntry(0,w,BMOPENII,8,w,r,b,x,r,w,r,w)
%TableEntry(0,w,BMCOMMAND,4,b,w,r,w)
%TableEntry(0,w,BMREPORT,2,b,x)
%TableEntry(0,w,BMREPORTWAIT,2,b,x)
%TableEntry(0,w,BMREPORTTIMEOUT,3,b,x,w)
%TableEntry(0,w,BMGETSTATUS,9,r,w,b,w,b,r,w,r,w)
%TableEntry(0,w,BMQUERYHANDLE,2,b,r)
%TableEntry(0,w,BMSETXLATMODE,3,b,b,b)
%TableEntry(0,w,SELECTBMULTI,2,r,w)
%TableEntry(0,w,SETXLATMODEBMULTI,3,b,b,b)
```

Add these lines if you are using COBOL release 5.0 or higher:

```
%ExtrnRtn(OPENBMULTI,w,0,5,w,b,w,x,x)
%ExtrnRtn(READBMULTI,w,0,3,b,x,w)
%ExtrnRtn(WRITEBMULTI,w,0,3,b,x,w)
%ExtrnRtn(SETOPTIONBMULTI,w,0,2,b,b)
%ExtrnRtn(RESETBMULTI,w,0,1,b)
%ExtrnRtn(CLOSEBMULTI,w,0,1,b)
%ExtrnRtn(BMOPENII,w,0,8,w,x,b,y,x,w,x,w)
%ExtrnRtn(BMCOMMAND,w,0,4,b,w,x,w)
%ExtrnRtn(BMREPORT,w,0,2,b,y)
%ExtrnRtn(BMREPORTWAIT,w,0,2,b,w)
%ExtrnRtn(BMREPORTTIMEOUT,w,0,3,b,y,w)
%ExtrnRtn(BMGETSTATUS,w,0,9,r,w,b,w,b,r,w,r,w)
%ExtrnRtn(BMQUERYHANDLE,w,0,2,b,r)
%ExtrnRtn(BMSETXLATMODE,w,0,3,b,b,b)
%ExtrnRtn(SELECTBMULTI,0,w,2,r,w)
%ExtrnRtn(SETXLATMODEBMULTI,0,w,3,b,b,b)
```

FORTRAN (ForGen.Asm)

Add these lines if you are using FORTRAN release 4.0 or lower:

```
%TableEntry(OPENBMULTI,BMULTO,5,w,b,w,r,r)
%TableEntry(READBMULTI,BMULTR,3,b,r,w)
%TableEntry(WRITEBMULTI,BMULTW,3,b,r,w)
%TableEntry(SETOPTIONBMULTI,BMULTS,2,b,b)
%TableEntry(RESETBMULTI,BMULTT,1,b)
%TableEntry(CLOSEBMULTI,BMULTC,1,b)
%TableEntry(BMOPENII,BMOPII,8,w,r,b,r,r,w,r,w)
%TableEntry(BMCOMMAND,BMCMND,4,b,w,r,w)
%TableEntry(BMREPORT,BMREPT,2,b,r)
%TableEntry(BMREPORTWAIT,BMWAIT,2,b,r)
%TableEntry(BMREPORTTIMEOUT,BMTIME,3,b,r,w)
%TableEntry(BMGETSTATUS,BMGETS,9,r,w,b,w,b,r,w,r,w)
%TableEntry(BMQUERYHANDLE,BMQRYH,2,b,r)
%TableEntry(BMSETXLATMODE,BMSXLT,3,b,b,b)
%TableEntry(SELECTBMULTI,SELTBM,2,r,w)
%TableEntry(SETXLATMODEBMULTI,SXLTBM,3,b,b,b)
```

Add these lines if you are using FORTRAN release 5.0 or higher:

```
%mediate(OPENBMULTI,BMULTO,5,w,b,w,r,r)
%mediate(READBMULTI,BMULTR,3,b,r,w)
%mediate(WRITEBMULTI,BMULTW,3,b,r,w)
%mediate(SETOPTIONBMULTI,BMULTS,2,b,b)
%mediate(RESETBMULTI,BMULTT,1,b)
%mediate(CLOSEBMULTI,BMULTC,1,b)
%mediate(BMOPEN,BMOPNI,8,w,r,b,r,r,w,r,w)
%mediate(BMOPENII,BMOPII,8,w,r,b,r,r,w,r,w)
%mediate(BMCOMMAND,BMCMND,4,b,w,r,w)
%mediate(BMREPORT,BMREPT,2,b,r)
%mediate(BMREPORTWAIT,BMWAIT,2,b,r)
%mediate(BMREPORTTIMEOUT,BMTIME,3,b,r,w)
%mediate(BMGETSTATUS,BMGETS,9,r,w,b,w,b,r,w,r,w)
%mediate(BMQUERYHANDLE,BMQRYH,2,b,r)
%mediate(BMSETXLATMODE,BMSXLT,3,b,b,b)
%mediate(SELECTBMULTI,SELTBM,2,r,w)
%mediate(SETXLATMODEBMULTI,SXLTBM,3,b,b,b)
```

Commands, Reports, and Requests

A BMULTI application instructs BMULTI through commands. BMULTI informs the application of events on the line by means of reports. Commands from an application may be accepted or denied by BMULTI depending on the state of the application's address. An address may move from one state to another in response to a command or an event on the line. Refer to Section 8, "BMULTI State Machine," for more information about BMULTI states.

The application issues commands and obtains reports by using procedural calls. The procedural calls issue operating system primitives called requests. These requests may either suspend themselves until a reply is available (Wait), or continue to process, checking periodically to see if a reply is available (Check).

Section 8
BMULTI State Machine

The current state of BMULTI affects the action it takes in response to a command from an application; therefore, it can be thought of as a state machine. The action depends on previous commands and events on the data communications line (which are passed to the application as reports). BMULTI may be in any one of nine states:

- Idle
- Offline
- Transmit Ready
- Transmitting
- Transmitting and Receive Ready
- Transmit and Receive Ready
- Receiving
- Receive Ready
- Receiving and Transmit Ready

Figure 8–1 shows the relationships between the states. BMULTI runs a parallel state machine for each address.

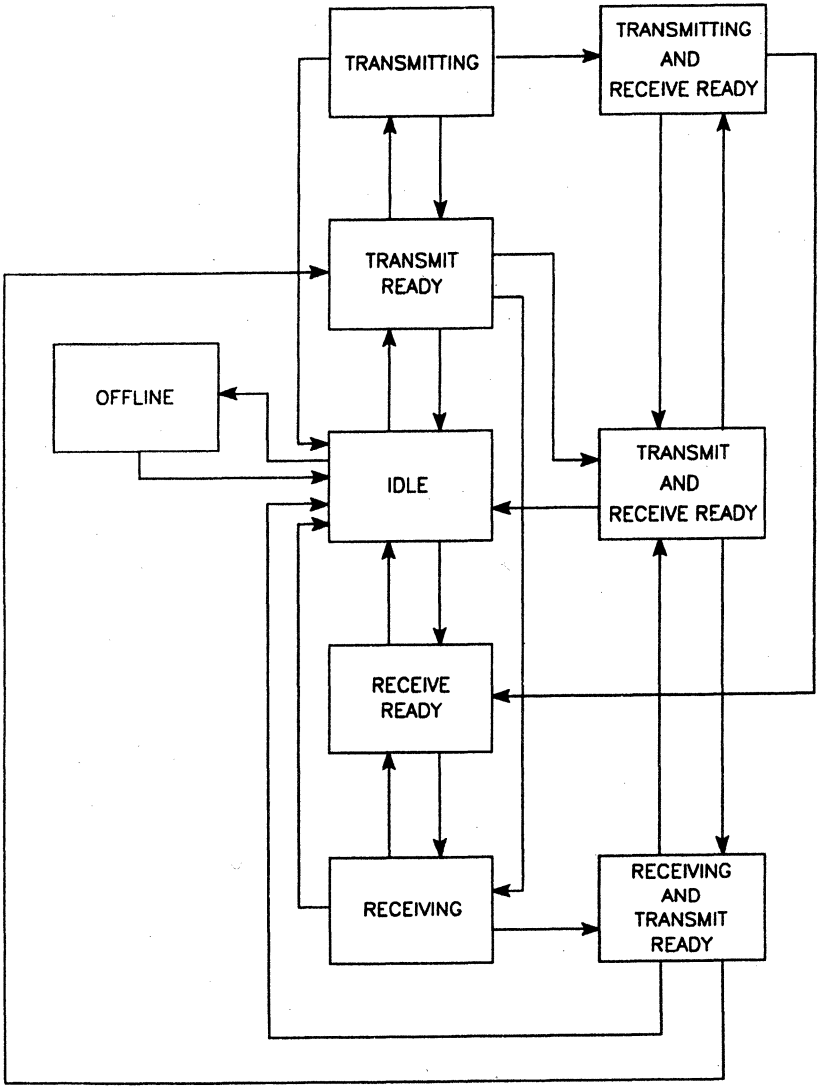

Figure 8–1. BMULTI States

The concepts used to explain the BMULTI states are:

- **Command Accepted and Command Denied.** "Accepted" indicates that BMULTI returns a Command Accepted code and performs the requested action. "Denied" means that it returns a Command Denied code and does not perform the requested action.
- **Fast Ready Flag.** The Set Fast Ready and Reset Fast Ready commands are accepted in all states except Offline. These two commands alter the setting of an internal flag that BMULTI uses to determine the appropriate response to fast, group, and broadcast selects.
- **End Session Command.** The End Session command is accepted from two states, Offline and Idle. The End Session command causes BMULTI to remove the associated address from the table of currently active addresses.

Reserving a Station Address

A Configure command (such as BmOpenII) must be made first to reserve a station address. Until the application system issues such a call, any attempts by the host computer to select or poll the station will be ignored by BMULTI (unless it is a Virtual Address (VAD)). BMULTI continues to ignore any such attempts until an Online command is issued. When a configure call is accepted by BMULTI, the application system is in the Offline state.

Online, Offline, and Idle

To move an application from the Offline state to the Idle state, use an Online command. When Idle, BMULTI replies to selects and polls addressed to the application station. BMULTI sends NAK in reply to selects and sends EOT in reply to polls. To return BMULTI to the Offline state, use the Offline command.

Virtual addresses (VADs) always behave as if in the Idle state.

Transmitting

The Transmit Ready command, when accepted, moves the application to the Transmit Ready state. In this state, BMULTI looks for the next poll to the application's address. Upon seeing such a poll, BMULTI issues a Ready for Transmit Buffer report. When returned, the application should immediately issue a Transfer Transmit Buffer command to instruct BMULTI to obtain and transmit the buffer. When BMULTI sees an ACK from the host, it returns a Transmit Done report.

If the application sees a Transmit Error report, it should look for another Ready for Transmit Buffer report and be prepared to issue another Transfer Transmit Buffer command.

Receiving

To move the application to the Receive Ready state, use the Receive command. In the Receive Ready state, BMULTI sends an ACK to any subsequent select operations, and receives the transmitted message. If BMULTI does not detect an error in the message, it issues a Receive Done report. The application program should immediately issue a Transfer Receive Buffer command to retrieve the buffer because BMULTI keeps the buffer for a limited period of time. If the command is accepted, an ETX occurs after the last text character in the buffer.

BMULTI issues the reports Receive Error, Duplicate Sequence Number, and Sequence Number Error (instead of Receive Done) when it receives a message in which it detects an error.

Idle and Abort

After a Receive or Transmit command is issued, you can use the Idle command to return the application to the Idle state. However, if a select has already been acknowledged (ACK) or a transmission begun in response to a poll, use Abort to cause BMULTI to abandon attempts to receive or to transmit a message.

For example, if a message sent to the workstation contains embedded ETXs, BMULTI will not acknowledge it. BmReport will return a value of 7. After several successive returns, the application should issue an Abort command and either warn the operator or end the session.

Receiving Fast, Group, and Broadcast Selects

If the Set Fast Ready command is not issued, BMULTI sends a NAK in reply to any fast, group, or broadcast selects addressed to the application's station except when Receive Ready. If the Set Fast Ready command is issued, BMULTI acknowledges and receives any such selects, even if it is not Receive Ready. (During the reception of a group or broadcast select, BmReport returns a value of 2 in order to distinguish between messages designated for that station in particular and messages designated for many stations.) After a Set Fast Select command is issued, a Reset Fast Select command may be issued to prevent reception of fast, group, or broadcast select.

Terminating a Communication Session

The application terminates a communication session by issuing an End Session command. This command is accepted from only the Idle and Offline states.

Report Queue

BMULTI maintains a 10–deep queue of reports for each active address. Report codes Receiving, Receiving Group or Broadcast Select, Select Denied, and Receive Error are added to the report queue only if they are not already in the queue. Keep the report queue as shallow as possible by frequently reading it; if the queue is too deep, reports returned by BMULTI may be obsolete.

BMULTI States

Note: In the following discussion, the commands Set Fast Ready, Reset Fast Ready, and End Session are not listed. The preceding discussion applies.

Offline

Offline is the initial state of any address when BMULTI accepts a Configure command. In this state, BMULTI ignores all control sequences for the assigned address. Only the Online and End Session commands are accepted; all others are denied.

Commands	State Change
Online	Idle
End Session	Exits the BMULTI state machine

Idle

In Idle state, BMULTI is not ready to receive any data but is responsible for responding for the address. However, fast, group, and broadcast selects are accepted if the Fast Ready flag is on.

In this state, the following commands are accepted:

Commands	State Change
Offline	Offline
Idle	None
Receive	Receive Ready
Transmit	Transmit Ready
Transfer receive buffer	None
Abort	None
End Session	Exits the BMULTI state machine

Inputs from the communications channel:

Poll of configured address:

State Change:	None
Report:	None
Action:	Transmit an EOT

Group poll of installed group poll addresses:

State Change:	None
Report:	None
Action:	If downstream RTS is FALSE and none of the other addresses assigned to this cluster are in Transmit Ready, transmit an EOT; otherwise, no response

Select of configured address:

State Change:	None
Report:	Select denied
Action:	Transmit a NAK

Fast Select of configured address with Fast Ready flag set:

State Change:	Receiving
Report:	Receiving
Action:	Wait for SOH

Fast Select of configured address with Fast Ready flag cleared:

State Change:	None
Report:	Select denied
Action:	Wait for the ETX and transmit a NAK

Broadcast Select of configured address or Group Select of configured address with installed Group Select character, with Fast Ready flag set:

State Change:	Receiving
Report:	Receiving group or broadcast select
Action:	Wait for SOH

Broadcast Select of configured address or Group Select of configured address with installed Group Select character, and Fast Ready flag cleared:

State Change: None
Report: Select denied
Action: Wait for the ETX and transmit a NAK

Broadcast Select of unconfigured address or Group Select of unconfigure address, with Fast Ready flag cleared:

State Change: None
Report: None

Broadcast Select of unconfigured address or Group Select of unconfigure address, with Fast Ready flag set:

State Change: Receiving
Report: Receiving group or broadcast select

Transmit Ready

In Transmit Ready state, BMULTI is ready to transmit and is not ready to receive any data. In this state the following commands are accepted:

Commands	State Change
Idle	Idle
Transfer receive buffer	None
Abort	Transmit and Receive Ready

Inputs from the communications channel:

Poll of configured address:

State Change: Transmitting
Report: Ready for transmit buffer
Action: Transmit message

Group poll of installed group poll addresses:

State Change:	Transmitting
Report:	Ready for transmit buffer
Action:	Block downstream RTS and CTS; transmit message for each online application that is Transmit Ready, one by one.

Select of configured address:

State Change:	None
Report:	Select denied
Action:	Transmit a NAK

Fast Select of configured address with Fast Ready flag set:

State Change:	Receiving and Transmit Ready
Report:	Receiving
Action:	Wait for SOH

Fast Select of configured address with Fast Ready flag cleared:

State Change:	None
Report:	Select denied
Action:	Wait for the ETX and transmit a NAK

Broadcast Select of configured address or Group Select of configured address with installed Group Select character, with Fast Ready flag set:

State Change:	Receiving
Report:	Receiving group or broadcast select
Action:	Wait for SOH

Broadcast Select of configured address or Group Select of configured address with installed Group Select character, and Fast Ready flag cleared:

State Change:	None
Report:	Select denied
Action:	Wait for the ETX and transmit a NAK

Broadcast Select of unconfigured address or Group Select of unconfigured address, with Fast Ready flag cleared:

State Change:	None
Report:	None

Broadcast Select of unconfigured address or Group Select of unconfigured address, with Fast Ready flag set:

State Change: Receiving
Report: Receiving group or broadcast select

Transmitting

In the Transmitting state, BMULTI recognized a poll or group poll and is ready to transmit data on the communications channel. The following commands are accepted:

Idle (if CTS is not on):

State Change: Idle
Action: Turn off RTS

Transfer transmit buffer:

State Change: None

Receive:

State Change: Transmitting and Receive Ready

Abort:

State Change: None
Action: Set Fast Ready to FALSE, turn off RTS

Inputs from the communications channel:

EOT:

State Change: Transmit Ready
Report: None
Action: Unblock downstream RTS and CTS

ACK:

State Change: Idle
Report: Transmit done
Action: If no more applications are Transmit Ready (for a group poll), unblock downstream RTS and CTS; if downstream RTS is FALSE, transmit EOT; for specific poll, transmit EOT

NAK:

State Change:	None
Report:	None
Action:	Retransmit the data according to the protocol

RVI:

If specific poll:

State Change:	Idle
Report:	Transmit done
Action:	Transmit an EOT

If station transmitted last in reply to a group poll:

State Change:	Idle
Report:	Transmit done
Action:	Unblock downstream RTS and CTS; if downstream RTS is FALSE, transmit an EOT; otherwise, no response

If station is waiting to be unblocked and has not had an opportunity to reply to the group poll:

State Change:	Transmit Ready
Report:	Transmit Error
Action:	Unblock downstream RTS and CTS; if downstream RTS is FALSE, transmit an EOT; otherwise, no response

Transmitting and Receive Ready State

In Transmitting and Receive Ready state, BMULTI recognized a poll or group poll and is ready to transmit data on the communication channel. The addressed workstation is also ready to accept data. When the transmission is complete, the workstation will be in the Receive Ready state.

The following commands are accepted:

Transfer transmit buffer:

State Change: None

Idle (if CTS is not on):

State Change: Idle
Action: Turn off RTS

Abort:

State Change: Idle
Action: Set Fast Ready to FALSE; turn off RTS

Inputs from the communications channel:

EOT:

State Change: Transmit and Receive Ready
Report: None
Action: Unblock downstream RTS and CTS

ACK (for client workstations individually, in the case of group poll):

State Change: Receive Ready
Report: Transmit done

NAK:

State Change: None
Report: None
Action: Retransmit the data according to the protocol

RVI:

If specific poll:

State Change: Idle
Action: Transmit an EOT

If station transmitted last in reply to a group poll:

State Change: Receive Ready
Report: Transmit done
Action: Unblock downstream RTS and CTS; if downstream RTS is FALSE, transmit an EOT; otherwise, no response

If station is waiting to be unblocked and has not had an opportunity to reply to the group poll:

State Change:	Transmit and Receive Ready
Report:	Transmit Error
Action:	Unblock downstream RTS and CTS; if downstream RTS is FALSE, transmit an EOT; otherwise, no response

Transmit and Receive Ready State

In the Transmit and Receive Ready state, the protocol handler is ready to transmit and receive data.

The following commands are accepted:

Idle:

State Change: Idle

Abort:

State Change: Idle

Inputs from the communications channel:

Poll of configured address:

State Change:	Transmitting and Receive Ready
Report:	Ready for Transmit Buffer
Action:	Block downstream RTS and transmit data

Group poll of configured group poll addresses:

State Change:	Transmitting and Receive Ready
Report:	Ready for Transmit Buffer
Action:	Block downstream RTS and CTS

Select of configured address:

State Change:	Receiving and Transmit Ready
Report:	Receiving

Fast select of configured address:

State Change:	Receiving and Transmit Ready
Report:	Receiving
Action:	Wait for SOH

Broadcast Select:

State Change:	Receiving and Transmit Ready
Report:	Receiving
Action:	Wait for SOH

Receiving State

In Receiving state, the protocol handler is receiving a block of data.

The following commands are accepted:

Abort:

State Change:	Idle
Action:	Set Fast Ready to FALSE; turn off RTS

Transmit:

State Change:	Receiving and Transmit Ready

Inputs from the communications channel:

EOT:

State Change:	Receive Ready
Report:	None

ETX (and Block Check Character; no parity or BCC error):

State Change:	Idle
Report:	Receive Done
Action:	Transmit an ACK if select was on this address

ETX (and Block Check Character; parity or BCC error):

State Change:	None
Report:	Receive Error
Action:	Transmit a NAK if select was on this address and wait for SOH or EOT

Receive Ready State

In the Receive Ready state, BMULTI is ready to receive data.

The following commands are accepted:

Idle:

State Change: Idle

Abort:

State Change: Idle

Transmit:

State Change: Receiving and Transmit Ready

Inputs from the communications channel:

Poll of configured address:

State Change: None
Report: None
Action: If downstream RTS is FALSE, transmit an EOT

Group poll of configured group poll addresses:

State Change: None
Report: None
Action: If downstream RTS is FALSE, transmit an EOT

Select of configured address:

State Change: Receiving
Report: Receiving

Fast Select of configured address:

State Change: Receiving
Report: Receiving
Action: Wait for SOH

Broadcast Select:

State Change: Receiving
Report: Receiving
Action: Wait for SOH

Group Select of installed Group Select character:

State Change:	Receiving
Report:	Receiving
Action:	Wait for SOH

Receiving and Transmit Ready State

In Receiving and Transmit Ready state, BMULTI is receiving a block of data and is also ready to transmit on the next poll with this workstation's address.

The following commands are accepted:

Idle:

State Change:	Idle

Abort:

State Change:	Idle
Action:	Set Fast Ready to FALSE

Inputs from the communications channel:

EOT:

State Change:	Transmit and Receive Ready
Report:	Receive Error

ETX (and Block Check Character; no parity or BCC error):

State Change:	Transmit Ready
Report:	Receive Done
Action:	Transmit an ACK if select was on this address

ETX (and Block Check Character; parity or BCC error):

State Change:	None
Report:	Receive Error
Action:	Transmit a NAK if select was on this address and wait for SOH or EOT

Section 9
Protocol Description

This section describes the multipoint terminal protocol. In this discussion, CTOS systems running BMULTI applications are considered to be terminals.

Figures 9–1 through 9–7, shown in this section, include all protocol options implemented in BMULTI. Most users need to know only a few of these.

Asynchronous Data Communication

In asynchronous data communication, each transmitted character uses ten nominally equal time intervals. The time intervals represent a start bit, eight bits of information, and a stop bit. Of the eight information bits, seven represent an ASCII character; the eighth is a parity bit.

Synchronous Data Communication

In synchronous data communication, each transmitted character uses eight nominally equal time intervals, representing eight bits of information. The first seven bits represent the seven bit character code, transmitted with the least significant bit first. The eighth bit is a parity bit selected to make the number 1, or marking bits of the 8–bit group odd. The next transmission character follows immediately.

Data Accountability

In transferring data from one point to another, proper accountability for each message is required under certain conditions. For example, in the handling of financial transactions, such as electronic transfer of funds, it is imperative that messages are not lost or duplicated. Where loss of a message or duplication is not important, transmission numbering may not be necessary. If each message sent has a transmission number serially assigned to it, the receiver can check that:

- Each message sent is received.
- Messages are received in the order they are sent.
- A message is not a retransmission of a previously transmitted message; therefore, it is not handled twice.

The message numbers sent from the host do not need to have a relationship to those sent from the other terminal and vice versa. Data transmission is not a balanced function; that is, one message sent does not always result in one reply.

Alternating Transmission Numbering

The minimum level of message numbering is a single character that alternates between an even and odd state. This system cannot distinguish between an error caused by message loss and an error caused by message duplication, though normal protocol procedures should prevent message loss. BMULTI allows for alternating 0 and 1, or for alternating @ and "A".

Sequential Transmission Numbering

Sequential transmission numbering provides more positive indication of message loss or duplication than the odd/even method. BMULTI allows one–, two–, or three–digit transmission numbers starting at 0 (or 00 or 000) and cycling through 9, 99, or 999.

No Response Timeout

The timing starts after transmission of a character signifying reversal of transmission direction. The time ranges from one to three seconds.

If the first character of a terminal transmission is not received, or if the character received is not valid in its time, the controller or terminal repeats its transmission *n* times, in which *n* is greater than or equal to 0. Then, if the same condition exists, it interrupts and enters the necessary error recovery procedures.

If the reversal is a result of an ACK or NAK, no repeat of the ACK or NAK is sent; however, EOT is sent to return to the control state.

Idle Line

The timing starts on receipt of each character other than a character signifying reversal of direction of transmission. Time ranges from one to three seconds. If the next character is not received in this time, the central processor interrupts and enters the necessary error recovery procedures.

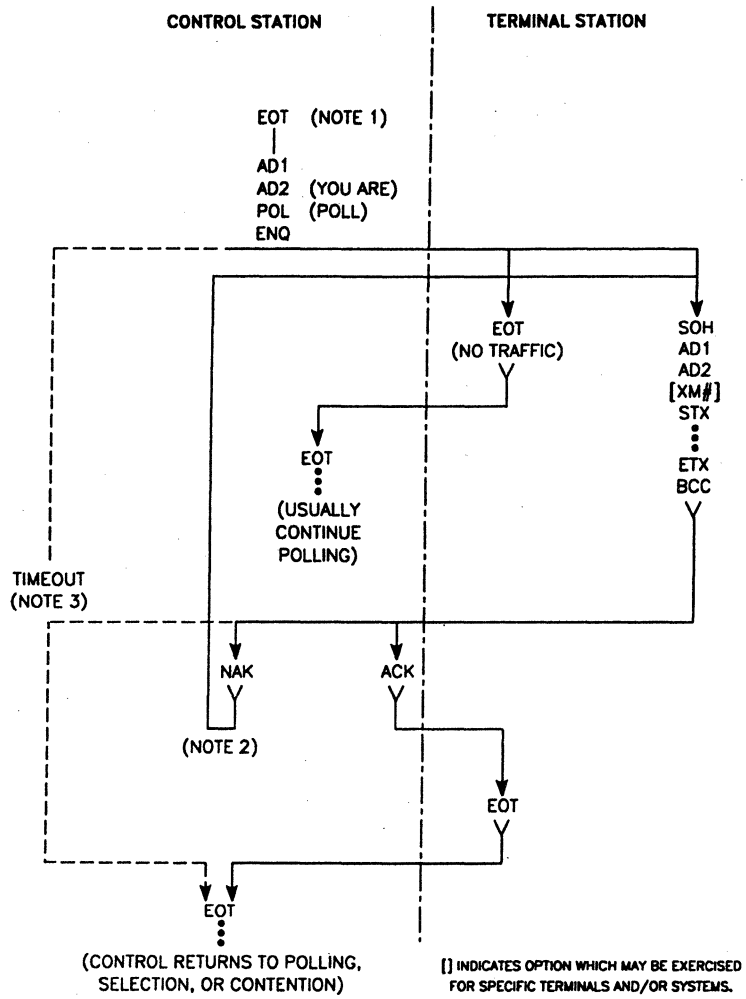

Figure 9–1. Specific Polling

Notes to Figure 9–1, Specific Polling

1. This EOT must come from the control station and may have been the termination of a previous transmission sequence. To minimize the effect of noise, the polling sequence may follow immediately.
2. If the control station receives a message for which character parity or block check test fails, NAK is transmitted, calling for a repeat of the transmission. This may be repeated *n* times (to be defined by the control station programmer), at which time, if the test fails, an error is recorded at the control station and EOT is transmitted, terminating the sequence. The terminal transmits the same message when next polled.
3. If the terminal does not receive ACK, NAK, or EOT, it may retain its message and remain quiet. The control station then times–out and transmits EOT, terminating the sequence. In this case, the message is retransmitted when next polled.

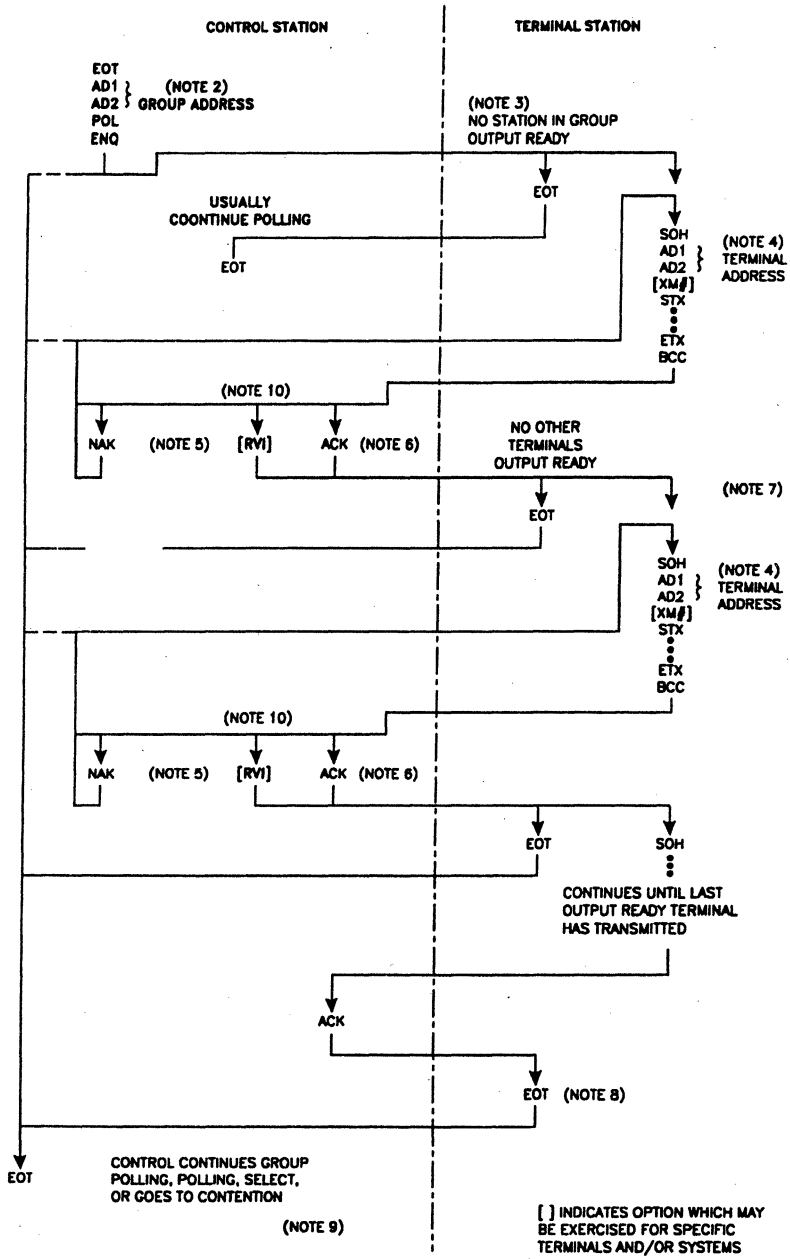

Figure 9–2. Group Polling

Notes to Figure 9–2, Group Polling

1. This procedure can reduce the overhead in a network of terminals in which several terminals are located at one location on a common communication line. The receipt of one group poll results in one response for the group, assuming no terminals are output ready. Thus, the control station can pass to the next group. During low activity, the control station can go through the polling list, noting the output status of all terminals with only one poll to each location, not each terminal. Also, if multiple terminals are output ready at a location, they are allowed to transmit, in sequence, in response to one poll. Selecting, broadcast, select, fast select, and so on are not affected by this group polling procedure.
2. The polling sequence follows the same format as a normal poll and uses the normal poll character. Group polling is controlled by address only. Terminals at a common location to be a part of a group are so identified by making their group poll addresses the same.
3. When the poll is received by the group addressed, the output ready terminals respond in the normal manner.
4. Each message sent in response to a group poll contains the address of the individual terminal that is responding.
5. If the control station detects an error in the message received in response to a group poll, normal polling error recovery is used.
6. The control station must, under this procedure, be certain that buffer space exists when it replies with an ACK to a message, or that the space will be available for the next message coming from another output ready terminal.
7. As soon as an ACK is received from the control station, the next output ready terminal transmits.
8. When an ACK is received from the control station and no terminals remain output ready, the last terminal to transmit is responsible to transmit the final EOT.
9. The same error recovery procedure outlined in Figure 9-1 is used with this procedure.
10. Reverse interrupt (RVI) may be used by the control station only after receiving a valid message that results in a positive acknowledgment. Instead of sending ACK, the control station sends RVI (DLEG).

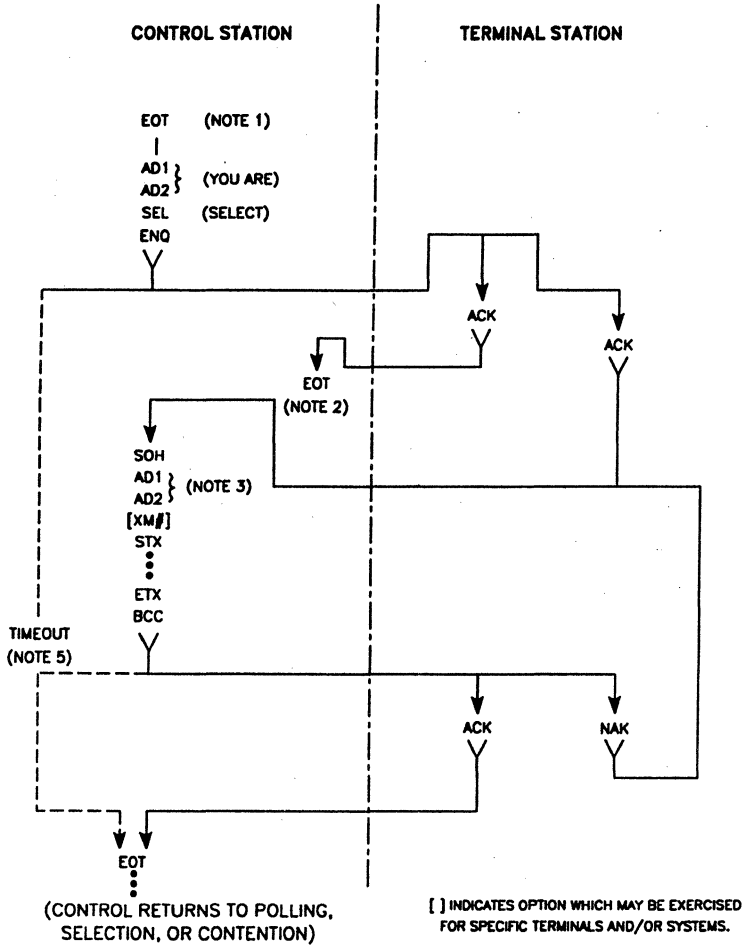

Figure 9–3. Selection

Notes to Figure 9–3, Selection

1. This EOT must come from the control station and may have been the termination of a previous transmission sequence. To minimize the effect of noise, the selection sequence may follow immediately.
2. If the terminal is not ready to receive, as indicated by transmission of a NAK, the control station normally retries the selection at the proper sequence of that terminal.
3. The identification characters in a transmission represent the terminal address for selection verification purposes. If the terminal fails to verify the address, it ignores the message.
4. If character parity or block check are not validated by the terminal, it sends a NAK. In this case, the control station retransmits the message *n* times (*n* may be equal to zero). If the terminal still does not acknowledge the message, the control station terminates the sequence with EOT, after recording the error. The control station retains the message for transmission on the next selection sequence to this terminal.
5. If the control station does not receive a response (ACK or NAK) to its message, it may timeout and retransmit the message *n* times (*n* may equal zero). If still no response is received, the control station terminates the sequence with EOT, after recording the error. The control station retains the message for transmission on the next selection sequence to this terminal.

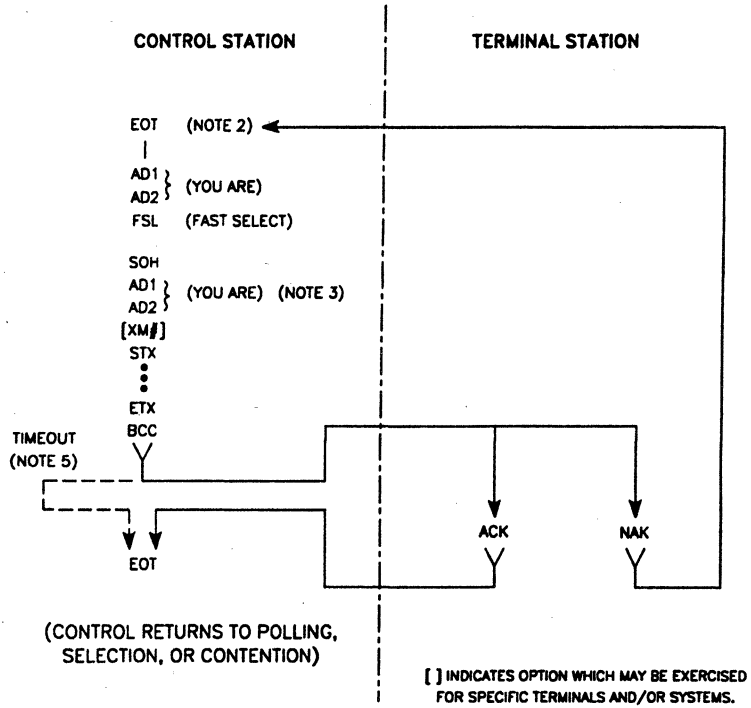

Figure 9–4. Fast Select

Notes to Figure 9–4, Fast Select

1. Fast selection is used when the control station wants to send a message to a terminal without first testing to make sure that the terminal is ready to receive. In this case, the selection and the message are transmitted together. The ACK response from the terminal applies both to the select and to a successful message transfer. A NAK response may indicate either that the terminal is not ready to receive or that the parity or block check in the message is invalid.
2. This EOT must come from the control station and may have been the termination of a previous transmission sequence. To minimize the effect of noise, the fast selection sequence may follow immediately.
3. The identification characters in a transmission from the control station also represent the terminal address "YOU ARE" for selection verification purposes. If either pair of addresses fail to verify, the terminal ignores the message.
4. If character parity or block check are not validated by the terminal selected, or if the terminal selected is not ready to receive the message, it responds with a NAK. In this case, the control station retransmits the fast select transmission *n* times (*n* may equal zero). If the terminal still does not accept the message, the control station terminates the sequence and retains the message for transmission on the next selection sequence to this terminal.
5. If the control station does not receive a response (ACK or NAK) to its transmission, it times–out and terminates the sequence. The control station retains the message for transmission on the next normal selection for this terminal.

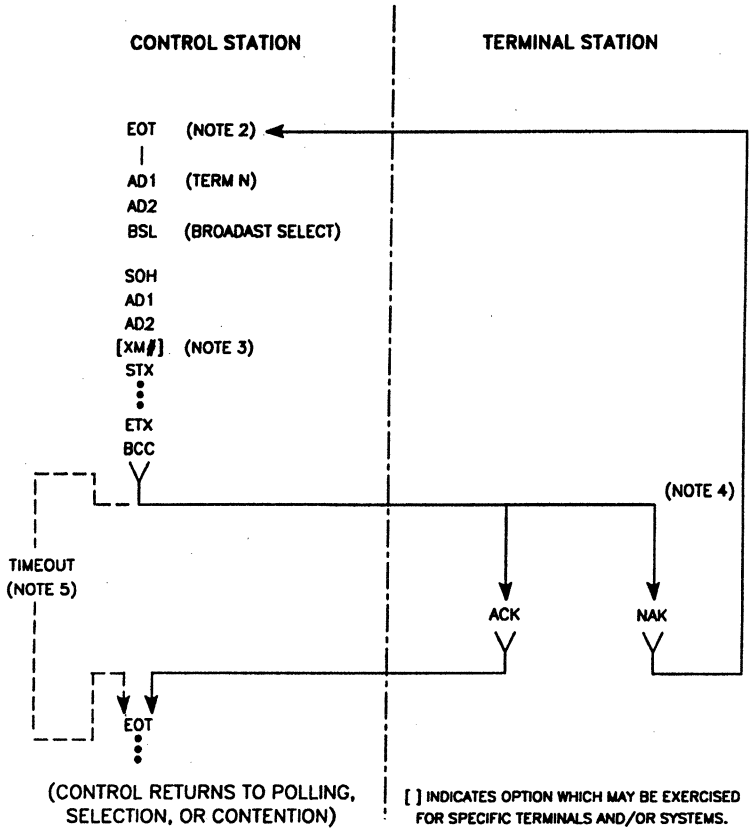

Figure 9–5. Broadcast Select

Notes to Figure 9–5, Broadcast Select

1. Broadcast select is a fast selection of all terminals. AD1–AD2 is selected to represent the terminal that acknowledges receipt of the message.
2. This EOT must come from the control station and may have been the termination of a previous transmission sequence. To minimize the effect of noise, the broadcast sequence may follow immediately.
3. Special sequences of numbers must be maintained if transmissions are numbered in a system in which broadcast is employed.
4. If the acknowledging terminal does not receive a valid message (for example, if there is a character parity or block check error) or is not receive ready, it transmits a NAK. The control station has the option of repeating the entire broadcast.
5. If the control station does not receive a response (ACK or NAK) to its broadcast, it may timeout and rebroadcast the message *n* times (*n* may equal zero). If no response is received, the control station terminates the broadcast mode with EOT after recording the error.

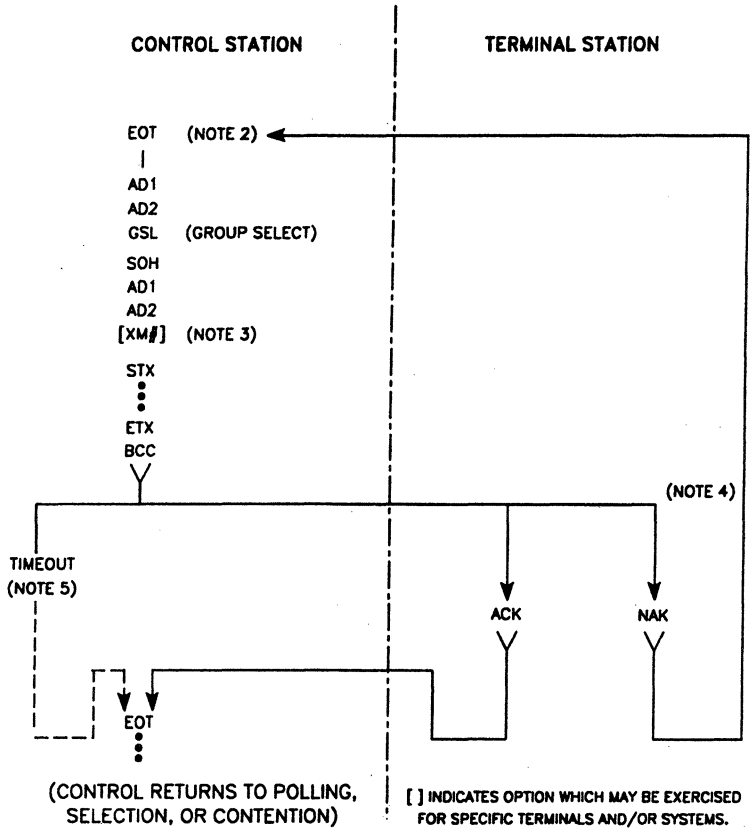

Figure 9–6. Group Select

Notes to Figure 9–6, Group Select

1. Group selection is a fast selection of a group of terminals. Each terminal can have a group select character for which it accepts a message. AD1–AD2 is selected to represent the address of the terminal that acknowledges receipt of the message.
2. This EOT must come from the control station and may have been the termination of a previous transmission sequence. To minimize the effect of noise, the group selection may follow immediately.
3. Special sequences of numbers must be maintained if transmissions are numbered in a system in which group selection is employed.
4. If the acknowledging terminal does not receive a valid message (for example, if there is character parity or block check error) or it is not receive ready, it transmits a NAK. The control station has the option of repeating the entire group selection.
5. If the control station does not receive a response (ACK or NAK) to the group selection, it may timeout and reselect the group *n* times (*n* may equal zero). If no response is received, the control station terminates the group select mode with EOT after recording the error.

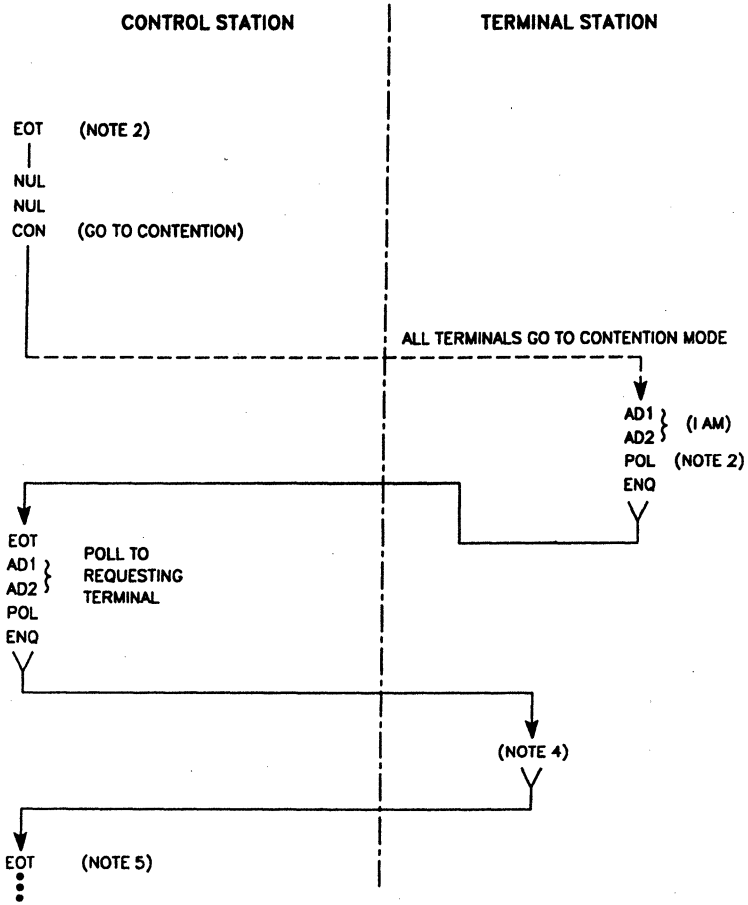

Figure 9–7. Multipoint Contention Mode

Notes to Figure 9–7, Multipoint Contention Mode

1. In times of low activity, it may be desirable to terminate polling and to place all or part of the system in the contention mode. This is done by transmission of *EOT NUL NUL CON* which causes the terminals to remain quiet until they have something to transmit.
2. This EOT must come from the control station and may have been the termination of a previous transmission sequence. To minimize the effect of noise, the multipoint contention sequence may follow immediately.
3. A terminal may wake up the polling activity by transmitting *AD1 AD2 POL ENQ*. This causes the control station to poll that terminal. If two terminals attempt to transmit at the same time, the garbled message initiates general polling by the control station.
4. The terminal proceeds with normal message transfer as in response to a poll (see Figure 9–1).
5. Following normal message receipt verification procedures, as in Figure 9–1, the control station may continue polling or instruct all terminals to go to contention.

Section 10
High–Level Procedural Interface

Of the interface levels provided by BMULTI, the High–Level Interface (HLI) is the easiest to use. It is therefore recommended that you use the HLI to create simple batch applications. See Appendix G for a sample program, a terminal emulator written in Pascal.

The procedures described in this section are:

- CloseBMULTI
- OpenBMULTI
- ReadBMULTI
- ResetBMULTI
- SelectBMULTI
- SetOptionBMULTI
- SetXlatModeBMULTI
- WriteBMULTI

The High–Level and Low–Level Interfaces support request routing over BNet and through Multiple Gateways Server (MGS). An application linked with either interface can access BMULTI running on a remote BNet node and/or under MGS.

The application status block (ASB) is used by the HLI to communicate the status of the reads and writes issued by an application. The format of the ASB is shown in Table 10–1.

Note: *If you are programming in Pascal, you should use explicit offsets when declaring this structure.*

Table 10–1. Application Status Block Format

Field	Size (Bytes)	Application Usage	Definition
RcvStatus	1	Read	Used by the HLI process to indicate Read status
RcvErc	2	Read	Internal error seen by the HLI process for receive
fSelDen	1	Read/Write/Flag	Set by the HLI process to TRUE if the line is selecting this address; an application should reset this flag to FALSE after processing
XmtStatus	1	Read	Used by the HLI process to indicate Write status
XmtErc	2	Read	Internal error seen by the HLI process for Xmt
Option	1	Read	Station option byte (used by SetOptionBMULTI)
fFMess	1	Read/Write/Flag	Set to TRUE if a fast select message is received by this address; an application should reset this flag to FALSE after processing
pFMess	4	Read	Address of fast message buffer (which is not in the application area); this pointer is returned by the HLI process after an OpenBMULTI)

CloseBMULTI

Description

CloseBMULTI frees the device address by issuing an end session command. This is normally the last step in a data communications session.

Procedural Interface

CloseBMULTI (Sh) : Erctype

Sh	A byte returned by OpenBMULTI.

OpenBMULTI

Description

OpenBMULTI opens a BMULTI station with the supplied device address and returns a station handle that should be used for all successive BMULTI calls. OpenBMULTI also creates the HLI process. The application must provide an ASB.

If using BMULTI with BNet and/or MGS, you must use the SelectBMULTI procedure to identify the copy of BMULTI you want to open.

Procedural Interface

OpenBMULTI (devAdr, fSys, Priority, pSh, pASBlk) : Erctype

DevAddr	A word containing two ASCII characters. This address must not be a duplicate of any other stations on the same line.
fSys	A byte or boolean value. It should be set to TRUE if the application making the call is to be a system service.
Priority	A word, containing the priority with which the HLI process will be created. This is normally higher than 128, for example, 127. (Priority is incremented in reverse order, with 1 being the highest priority.)
pSh	A pointer to a byte returned by OpenBMULTI.
pASBlk	A pointer to the application–supplied ASB. The format of the ASB is shown in Table 10–1.

ReadBMULTI

Description

ReadBMULTI initiates a receive operation. It does not wait for the message to arrive. The application should sample the ASB RcvStatus to determine when the receive is complete. When the receive is complete, the HLI process returns the received message in the receive buffer. The first two bytes contain the length of the received message.

You are responsible for creating a receive buffer of adequate size. If the buffer overflows, reported by the ASB when the RcvStatus field equals 14, you should issue a delay of approximately 1 second or issue a ResetBMULTI prior to issuing another ReadBMULTI.

Procedural Interface

ReadBMULTI (Sh, pBuf, sBuf) : Erctype

Sh	A byte (returned by OpenBMULTI).
pBuf	A pointer to an application–supplied buffer into which the received data will be placed.
sBuf	A word containing the maximum number of bytes the application can receive.

Possible hexadecimal values of RcvStatus:

00	Initial state or after a reset
01	Busy — read initiated
02	Read failure
10	Read complete — no errors
11	Read complete — transmission number error
12	Read complete — duplicate transmission number
14	Read complete — truncated message

ResetBMULTI

Description

ResetBMULTI restores the application status to idle and aborts an ongoing read or write. After a reset, the Option byte is set to FALSE.

Procedural Interface

ResetBMULTI (Sh) : Erctype

Sh	A byte (returned by OpenBMULTI).

SelectBMULTI

Description

The SelectBMULTI allows the user to select which copy of BMULTI to use to transfer data. The copy chosen can be on another node in a BNet network, and/or one that resides on one of multiple copies on the same system in the case of Multiple Gateways Server (MGS).

If no remote node (for BNet) or Service ID (for MGS) is named, the local copy of BMULTI is used. A node name and Service ID can be specified if MGS and BNet are both used.

The SelectBMULTI call must be made immediately before each OpenBMULTI to be effective, because the node specification entered as part of the SelectBMULTI call is stored for use by the OpenBMULTI procedure. Since this specification is used by each OpenBMULTI call that follows, the SelectBMULTI procedure should be used before each OpenBMULTI procedure. The use of SelectBMULTI is optional. If it is not used, the Service ID and node selection default to the local copy of BMULTI running outside of MGS.

SelectBMULTI is an object module procedure.

Procedural Interface

SelectBMULTI (pbNodeName, cbNodeName): Erctype;

pbNodeName, cbNodeName	A string that defines the location of the BMULTI to be used. The string specifies both BNet node name and MGS Service ID. Examples: {Center}, <Accounting>, {NYC}<Main>. The combined node name and Service ID can be no longer than 50 characters.

SetOptionBMULTI

Description

SetOptionBMULTI enables the application to select the BMULTI options: offline, online, fast ready, or not fast ready. Refer to Section 8, "BMULTI State Machine," for more information about these options.

Procedural Interface

SetOptionBMULTI (Sh, Option) : Erctype

Sh — A byte (returned by OpenBMULTI).

Option — A byte with values as follows:

Bit 0 is 1 = Offline
is 2 = Online

Bit 1 is 1 = Fast Ready
is 2 = Not Fast Ready

The ASB option field is updated to reflect the option selected.

SetXlatModeBMULTI

Description

SetXlatModeBMULTI is used to select the character translation algorithm to be used with extended character sets. This procedure works the same way as the BmSetXlatMode procedure included in the Low–Level Interface.

The commands to open BMULTI (all interfaces) establish the default mode as 0 (no translation required). Therefore, using of this procedure is optional. SetXlatModeBMULTI can be used to start, terminate, or change the existing character translation mode. Although the mode is individually selectable for each station, the translation table specified in the configuration file is common for all users of this copy of BMULTI.

To use this procedure, BMULTI must be offline, or else in the idle state and not fast–ready. If these conditions are not met, or if an invalid mode is specified, error code 32806 is returned.

SetXlatModeBMULTI is an object module procedure.

Procedural Interface

SetXlatModeBMULTI (Sh, iRcvMode, iXmitMode): Erctype;

Sh	A byte (returned by OpenBMULTI).
iRcvMode	Selects the character translation algorithm to be used for receiving characters. This parameter provides the same functionality as the iRvcMode described for the BmSetXlatMode procedure.
iXmitMode	Selects the character translation algorithm to be used for transmitting characters. This parameter provides the same functionality as iXmitMode described for the BmSetXlatMode procedure.

WriteBMULTI

Description

WriteBMULTI initiates a transmit operation, but does not wait for the message to be successfully transmitted. The application should sample the ASB XmtStatus to determine when the transmission is complete.

Procedural Interface

WriteBMULTI (Sh, pBuf, sBuf) : Erctype

Sh	A byte (returned by OpenBMULTI).
pBuf	A pointer to an application–supplied buffer that contains the data BMULTI will transmit.
sBuf	A word containing the number of bytes BMULTI will transmit.

Possible hexadecimal values of XmtStatus:

00	Initial state or after a reset
01	Busy — write initiated
02	Write failure
10	Write complete

Section 11
Low–Level Procedural Interface

This section describes the Low–Level Interface (LLI). The LLI is useful for applications that require extra control. In addition, you must use the LLI for applications to have access to the BMULTI Suspend/Resume filter.

The procedures described in this section are:

- BmCommand
- BmGetStatus
- BmOpenII
- BmReport
- BmReportTimeout
- BmReportWait
- BmResume
- BmSetXlatMode
- BmSuspend

The High–Level and Low–Level Interfaces both support request routing over BNet and through Multiple Gateways Server (MGS). A application linked with either interface can access BMULTI running on a remote BNet node and/or under MGS. Refer the *BTOS Systems BNet Administration Guide* and the *BTOS Multiple Gateways Server (MGS) Operations and Programming Guide* for more information.

BmCommand

Description

BmCommand passes commands from the application to BMULTI.

BmCommand returns ErcOK(0) without a wait if the command was a Transfer Transmit Buffer. BmCommand checks whether the request block returned was the one made in the request issued prior to the Transfer Transmit Buffer request. If it is the Transfer Transmit Buffer request, however, BmCommand performs another wait.

Procedural Interface

BmCommand (Sh, Command, pBuf, sBuf) : Erctype

Sh	The station handle, a byte returned by BmOpenII or BmResume.
Command	A word containing a value from Table 11-1.
	Note: *If BmCommand [End Session] is sent for a Suspend/Resume session, while that session is in the suspended state, the session is* not *cleared. For a non–Suspend/Resume session, however, the session is cleared when BmCommand [End Session] is called.*
pBuf	A pointer to an application–supplied message buffer into which data is to be transmitted from or placed upon receiving it (depending on the Command issued).
sBuf	These are normally dummy values except for the three buffer transfer commands: Get Receive Buffer, Send Transmit Buffer, and Transmit Extended message.

For Get Received Buffer, sBuf should be set to the maximum number of bytes that the application can accept. After BmCommand returns, the first two bytes starting at pBuf contain the number of bytes received.

For the Send Transmit Buffer or Transmit Extended Message, sBuf should be set to the number of bytes the application wants to transmit.

Table 11–1. BmCommand Command Parameter Values

01	Get Receive Buffer
02	Send Transmit Buffer
03	Offline
04	Online
05	Idle
06	Set Fast Ready
07	Set Receive Ready
08	Set Transmit Ready
09	End Session
10	Abort
11	Reset Fast Ready
12	Transmit Extended Message (Messages >2048 bytes)

BmGetStatus

Description

BmGetStatus provides the caller with release–independent access to BMULTI information. It consolidates the necessary data returned by th BmQuery and BmIdentify routines of the 4.0 and 5.0 releases of BMULTI and presents this data in condensed, usable formats. BmQuer and BmIdentify are not supported in BMULTI 7.0 and higher.

Procedural Interface

BmGetStatus (pbHostName, cbHostName, iStation, nStation, iRecordLen, pbCommonRet, cbCommonRet, pbStatRecsRet, cbStatRecsRet): Erctype;

pbHostName, cbHostName	The BNet node name of the copy of BMULTI to be used and its size. This field also contains the MGS Service ID, if applicable. The format should be in the form: *{Node}<ServiceID>*. The node name, if present, must be enclosed in braces; the Service ID if present, must be enclosed in angle brackets. If cbHostName is zero, the application is connected to the default route for BMULTI requests (BMULTI on the local node and running without MGS).
iStation	Index of the first station for which information will be returned.
nStation	Number of station records required.
iRecordLen	Length of each data record to be returned. If the total length of all records to be returned (iRecordLen * nStation) is greater than the length of the buffer specified in cbStatRecsRet, an error (ercInvalidRecordLength) is returned.

pbCommonRet, cbCommonRet	Description of the Common Information Buffer, where the general information of the same environment is returned. See Table 11-2 for the format of the Common Information Buffer.
pbStatRecsRet, cbStatRecsRet	Description of the Station Record Buffer, where station status records are returned. See Table 11–3 for the format of the Station Record Buffer.

Table 11–2. Common Information Buffer Format

Field	Offset	Size (bytes)	Comment
fLineActivity	0	2	Counter incremented by all transmit and receive interrupts.
WsNumber	2	2	Workstation number.
Channel	4	2	Two characters that specify which port the channel is using.
NakCount	6	2	NAK counter for this copy of BMULTI; the maximum count is 65535.
nActive Stations	8	2	Number of active stations; maximum 64.
Mode	10	1	*A* if asynchronous or *S* if synchronous.
BMULTIVersion	11	13	Installed BMULTI version format. The string length field is 1 byte. The character string field is a maximum of 12 bytes. The actual character string can be a maximum of 11 characters. The string is terminated by a null, which is used as an end–of–string delimiter in some languages. The null character is not counted when determining string length.

Table 11–3. Station Record Buffer Format

Field	Offset	Size (bytes)	Comment
StationIndex	0	2	The index of the station record.
DeviceAddr	2	2	The BMULTI line address.
UserNum	4	2	The application user number, as seen by BMULTI.
State	6	1	The current state of the station, where: 0 = Offline 1 = Idle 2 = Receive Ready 3 = Receiving 4 = Transmit Ready 5 = Transmit and Receive Ready 6 = Receiving and Transmit Ready 7 = Transmitting 8 = Transmitting and Receive Ready
Rqs	7	1	The number of outstanding requests.
Xmit	8	12	A 4–record array of 3–byte text strings. The records contain the expected transmission numbers for transmit, receive, group select, and broadcast select.
TxMsg	20	2	The count of transmitted messages.
RxMsg	22	2	The count of received messages.
StartTime	24	4	The date and time when the user first logged onto BMULTI.
ApplDesc	28	Up to 10	The user application that passes in the open command. Format: String length (1 byte), character string (up to 9 bytes). The actual character string is only 8 characters long. The string is terminated with a null character which is not included when determining the string length.
UserName	38	Up to 10	The application user name, with a string length of 1 byte and a character string of maximum 8 bytes, followed by a null character. The null character is not counted when determining string length.

BmOpenII

Description

Either BmOpenII or BmResume are generally the first procedures called by BMULTI applications.

With BmOpenII, you can request that an unused virtual address (VAD) be used instead of a specific address. (See Section 4, "Configuring and Installing BMULTI," for more information about VADs.)

BmOpenII also provides support for BNet (LANs) and multiple copies of BMULTI per system using Multiple Gateways Server (MGS). If you provide a node name (for BNet) and/or Service ID (for MGS), the node name and Service ID are used to route the request to the desired gateway. However, if a BMULTI application wishes to make use of the BMULTI Suspend/Resume filter, it must use the BmResume call instead.

It is important to replace BmOpen with BmOpenII before linking older applications with BMULTI. If this is not done, the link will fail.

Notes:

1. *Applications that call BmOpenII with the "any open VAD" option must be introduced very carefully into existing installations, especially if they are being mixed with older applications that have fixed addresses. See Appendix A, "Status Codes and Troubleshooting," for more details.*

2. *To use the BMULTI Suspend/Resume filter in an application, you must use the BmResume call to initiate a BMULTI gateway connection. If you use the BmOpenII call, the connection request bypasses the Suspend/Resume filter, and therefore the station address is not registered with the filter.*

Procedural Interface

BmOpenII (DevAddr, pShRet, fSys, pDevAdrRet, pbHostName, cbHostName, pbDescString, cbDescString) : Erctype

DevAddr	A word containing two ASCII characters, DevAdd denotes the address used by the host for polls and selects. This address must not be a duplicate of any other stations on the same line. If this word contains 0FFFFh, the first unused VAD is selected. If the word contains any other value, that value is used.
pShRet	A pointer to a byte allocated by the application, into which the BMULTI station handle is returned
fSys	A byte or boolean value. This flag determines whether transmission requests for this user number are to be honored. It should be set to TRUE (0FFh) if the application making the call is to be a system service, or FALSE (0h) otherwise.
pDevAdrRet	A pointer to a word allocated by the application. The value of the actual virtual address (VAD) allocated is returned here.
pbHostName, cbHostName	The BNet node name of the copy of BMULTI to be used and its size. This field also contains the MGS Service ID, if applicable. The format should be in the form: *{Node}<ServiceID>*. The node name, if present, must be enclosed in braces; the Service ID if present, must be enclosed in angle brackets. If cbHostName is zero, the application is connected to the default route for BMULTI requests (BMULTI on the local node, not under MGS).
pbDescString, cbDescString	A string (maximum length of 8 bytes) that describes the application (for example, "PPT", "HostLink", "Infoview", "BTE", "DTS"). If cbDescString is zero, the BMULTI status monitor will not list the application name.

BmReport

Description

BmReport returns the status of the data communications subsystem to the application.

If Erctype zero is returned, either there are no errors or no report is available. If a value other than zero is returned, the report is not valid.

Procedural Interface

BmReport (Sh, pReport) : Erctype

Sh — The station handle, a byte returned by BmOpenII or BmResume.

pReport — A pointer to a word into which the procedure is to return the report. Refer to Table 11-4 for pReport routing values and the procedures to which reports are routed.

Table 11–4. pReport Procedure Routing

0	No report
1	Receiving
2	Receiving Group or Broadcast Select
3	Receive Done
4	Ready for Transmit Buffer
5	Transmit Done
6	Select Denied
7	Receive Error
8	Duplicate Transmission Number
9	Transmission Number Error
10	Transmit Error

BmReportTimeout

Description

BmReportTimeout waits for a report for a specified period of time. The report is valid only if the value zero is returned.

Procedural Interface

BmReportTimeout (Sh, pReport, Timeout) : Erctyp

Sh	The station handle, a byte returned by BmOpenII or BmResume.
pReport	A pointer to a word into which the procedure will return the report. Refer to Table 11-4 for pReport routing values and the procedures to which reports are routed. BmReportTimeout returns the same reports as BmReport.
Timeout	A word that gives an interval (in tenths of a second) during which the procedure waits for a report.

BmReportWait

Description

BmReportWait waits for a report indefinitely. The report is valid only if the procedure returns zero.

Procedural Interface

BmReportWait (Sh, pReport) : Erctype

Sh	The station handle, a byte returned by BmOpenII or BmResume.
pReport	A pointer to a word into which the procedure will return the report. Refer to Table 11-4 for pReport routing values and the procedures to which reports are routed. BmReportWait returns the same reports as BmReport.

BmResume

Description

BmResume and BmSuspend are the LLI application programmatic interface to the BMULTI Suspend/Resume filter. The BmResume call enables an application to initiate a Suspend/Resume session or to assume control of a session in the suspended state.

To allow an application to use the Suspend/Resume filter, you must use the BmResume call instead of the BmOpenII call. Applications that use the BmOpenII call to establish connections bypass the filter and cannot be suspended or resumed. Refer to Section 12, "Using the BMULTI Suspend/Resume Filter," for more information.

The BmResume call is filtered by the Suspend/Resume filter. If the session to be resumed by the application is not yet established, the filter attempts to establish the connection by issuing a BmOpenII call to the BMULTI gateway. If the BmOpenII call is successful, the filter registers the station handle and passes it back to the application. If the address is already being used by a non–Suspend/Resume application, BMULTI responds with error code 32772, *Invalid Address*, and this error code is passed by the filter back to the application.

If the session for the indicated BMULTI address (**DevAddr**) is in the suspended state, the filter passes the registered station handle back to the application. The application then uses the station handle for communications on the established link.

If some other Suspend/Resume–registered application is currently using the requested address, the filter responds to the application with error code 60261, *Session Active*. If the BmResume call is received, and all configured Suspend/Resume sessions are in use, the filter responds with error code 60262, *Suspend/Resume session overflow*.

Procedural Interface

BmResume (DevAddr, pShRet, fSys, pDevAddrRet, pbHostName, cbHostName, pbDescString, cbDescString) : Erctype

DevAddr	A word indicating the device address, two ASCII characters. Applications using the same device address share the same session. The address used can be for any valid BMULTI address, except for BMULTI Virtual Addresses (VADs); therefore addresses in the range 20h < *DevAddr* < 7Fh can be used, while VADs, for example, 0FFFh, cannot be used as Suspend/ Resume addresses. If an application attempts to use a VAD as a Suspend/ Resume address, the filter responds with error code 60263, VADs not Supported.
pShRet	A pointer to a byte allocated by the application, into which the BMULTI station handle is returned.
fSys	A byte, set to TRUE (0FFh) if the application is a system service. This flag determines whether transmission requests for this user are to be honored.
pDevAddrRet	A pointer to a word allocated by the application. The value of the actual device address is returned here.
pbHostName, cbHostName	The BNet node name of the copy of BMULTI to be used and its size. This field also contains the MGS Service ID, if applicable. The format should be in the form: *{Node}<ServiceID>*. The node name, if present, must be enclosed in braces; the Service ID, if present, must be inclosed in angle brackets. If cbHostName is set to zero, the application is connected to the default route for BMULTI requests (BMULTI on the local node and running without MGS).

pbDescString, cbDescString	A string (maximum length, 8 bytes) that describes the application. Examples include: "PPT", "HostLink", "Infoview", "BTE", "DTS", and so on. If cbDescString is set to zero, the BMULTI status monitor will not list the application name.

BmSetXlatMode

Description

BmSetXlatMode is used to select the character translation algorithm to be used with extended character sets.

The commands to open BMULTI (all interfaces) establish the default mode as 0 (no translation required). Therefore, the BmSetXlatMode procedure is optional. It can be used to start, stop or change the existing character translation mode.

Although the mode is individually selectable for each station, the translation table loaded for the configuration file is common for all users of this copy of BMULTI. Examples are found in Appendix D, with translation tables in Appendix E.

To avoid errors, BMULTI must be offline or in the idle state and not fast–ready when using BmSetXlatMode.

Procedural Interface

BmSetXlatMode (Sh, iRcvMode, iXmitMode): Erctype;

Sh	The station handle, a byte returned by BmOpenII or BmResume.
iRcvMode	The character translation algorithm to be used for received characters. Enter *0* for no translation (all characters pass through). Enter *1* if the ESC SO escape sequence indicates that a string of encoded extended characters follows. The ESC SI escape sequence indicates that a string of standard (not extended) characters follows. When this mode is selected, these escape sequences are removed from the data stream when received.

Enter *2* if the ESC SO and the SO character alone each serve as indicators that a string of encoded extended characters follows. The ESC SI and SI character alone each indicate that a string of standard (not extended) characters follows. When this mode is selected, these sequences or characters are removed from the data stream when received.

Enter *4* if there is to be no decoding of the data stream. All characters in the message received will be translated using the translation table.

Enter *5* if the ESC SO escape sequence indicates that a string of encoded extended characters follows. The ESC SI escape sequence indicates that a string of standard (not extended) characters follows. When this mode is selected, these escape sequences are removed from the data stream when received. In addition, all characters received are translated after any necessary decoding is accomplished.

Enter *6* if the ESC SO and the SO character alone each serve as indicators that a string of encoded extended characters follows. The ESC SI and SI character alone each indicate that a string of standard (not extended) characters follows. When this mode is selected, these sequences or characters are removed from the data stream when received. In addition, all characters received are translated after any necessary decoding occurs.

iXmitMode

Selects the character translation algorithm to be used for transmitting characters.

Enter *0* for no translation. All characters pass through. The high–order bit of the byte value of any extended character is stripped away in the transmission process.

Enter *1* to insert the sequence ESC SO into the data stream to prefix a string of encoded extended characters, and to insert the sequence ESC SI to prefix a string of standard characters.

Enter *2* to insert only the SO character into the data stream to prefix a string of encoded extended characters. Only the SI character will be inserted into the data stream to prefix a string of standard characters.

Enter *4* if there is to be no encoding of the data stream. All characters in the message will be translated prior to transmission. The high–order bit of the byte value of any extended character will be lost in the transmission process.

Enter *5* to insert the sequence ESC SO into the data stream to prefix a string of encoded extended characters, and to insert the sequence ESC SI to prefix a string of standard characters. All characters in the message will be translated before determining whether encoding is necessary.

Enter *6* to insert only the SO character into the data stream to prefix a string of encoded extended characters. Only the SI character will be inserted into the data stream to prefix a string of standard characters. All characters in the message will be translated before determining whether encoding is necessary.

BmSuspend

Description

BmSuspend and BmResume are the LLI application programmatic interface to the BMULTI Suspend/Resume filter. The BmSuspend call enables an application to suspend control of an active BMULTI session.

To suspend a session, the session address and station handle must have been previously registered with the Suspend/Resume filter when the session was started. If the session was originally started through the BMULTI LLI, this means that the application starting the session must use the BmResume call instead of the BmOpenII call. Applications that use the BmOpenII call to establish connections bypass the filter and cannot be suspended or resumed. Refer to Section 12, "Using the BMULTI Suspend/Resume Filter," for more information.

The BmSuspend call is filtered by the Suspend/Resume filter. If the BmSuspend call is successful, all future requests from the application will receive the error code 60266, *Application Suspended.*

The exceptions to these future requests are for the BmResume and BmCommand [End Session] calls. If BmResume is received by the filter, it will attempt to resume the suspended session indicated by the byte containing the session address. When in the suspended state, the BmCommand [End Session] call can also be used to terminate the connection gracefully, without losing the session.

If the BmSuspend call cannot be accepted, for example, when the application state is "Receive Ready," "Receiving," or "Transmitting," the Suspend/Resume filter responds back to the application with error code 60260, *Application Cannot Be Suspended.*

If a non–Suspend/Resume application—one that did not initiate the original connection by using BmResume instead of BmOpenII—sends a BmSuspend call, the filter responds with error code 60264, *Invalid Command.*

Procedural Interface

BmSuspend (Sh) : Erctype

Sh	The station handle of the session address to be suspended, a byte returned by BmResume when the connection was established.

Section 12
Using the BMULTI Suspend/Resume Filter

This section describes the BMULTI Suspend/Resume filter, a BMULTI feature that allows different BMULTI applications to share the same station address. Applications sharing the same address therefore the same host session, although only one application can have session control at any given time.

The BMULTI Suspend/Resume filter allows multiple BMULTI applications to use the same BMULTI data communications address without having to finish out of the current host session. When you want to pass control of a given BMULTI address to another application, you *Suspend* the current session. For example, BTE 3.1 can share the same address with Finesse, and other BMULTI applications that have been enhanced to support Suspend/Resume operations.

Only one application at a time, however, can have an active session on a given station address. The Suspend/Resume filter works by allowing an application to voluntarily relinquish control of the active link, that is, *Suspend* the session. Once suspended, another BMULTI application or even the same application at later time can issue a *Resume* command to assume control of the session.

The suspended session can be resumed from any other BMULTI application that supports Suspend/Resume and is using the same address. The suspended session can also be resumed by another instance of the same application running on the same workstation (using Context Manager) or even a copy of the application running on a different workstation. An application can resume a suspended session, regardless of whether or not that application originally initiated the connection, as long as the application is configured to use the suspended address.

Any number of applications can share the same host session, even multiple instances of the same application (for example, using multiple copies of a terminal emulator program within Context Manager). A requirement for sharing is that the application must support the

BMULTI Suspend/Resume feature and use the Suspend/Resume interface. To share a session, the sharing applications must also be configured to use the same station address.

The Suspend/Resume filter can support up to a maximum of 128 sessions; the precise maximum number is configurable when the filter is installed as a system service.

For user–written applications, the BMULTI Low–Level Interface provides access to the Suspend/Resume filter through the BmResume and BmSuspend procedure calls. Refer to Section 11, "Low–Level Procedural Interface," for more information about writing applications to use the Suspend/Resume features.

Installing the Suspend/Resume Filter

The Suspend/Resume filter system service must be installed after the BMULTI gateway is installed. This applies for both manually installing the system services through the CTOS Executive command line or through the Sysinit.Jcl file at boot–up time.

To use the Suspend/Resume filter, you must be using BTOS II 3.0 or higher for CTOS workstations, CTOS/XE 3.0 or higher for shared resource processors, and IDMSS 4.0 or higher for the IDS and enhanced IDS modules. BMULTI 10.1 or higher (or BMULTI/X.25 1.1 or higher) must be installed and executing as a system service on the same CPU where the filter is to be installed.

BMULTI Virtual Address (VAD) support is not provided for the Suspend/Resume filter.

Manual Installation on a CTOS Workstation

1. Type *Install BMULTI S/R Filter* and press the RETURN key.
2. Enter the information in the form that appears

```
Install BMULTI S/R Filter
  Number of Sessions
```

In the **Number of Sessions** field, you must enter an integer number from 1 to 128. A session is one connection to a host. For example, if two applications are using the same address to communicate with a host, sharing control of communications through the Suspend/Resume filter, they are using the same session. The applications sharing the

address can share the host session, but only one application has control of the session at a time.

If you attempt to install the filter using a number outside the range of 1 to 128, the installation will fail and an error message will be displayed.

3. When you are finished entering the information, press the GO key.

Manual Installation on the IDS Module

As an alternative to installing BMULTI and the Suspend/Resume filter system services on the CPU of a CTOS workstation, you can instead install them on an IDS or enhanced IDS module attached to the workstation. When running on the IDS module, the CPU processor overhead for BMULTI and the filter is considerably lower than when running on the CPU module.

Note: *To install BMULTI on the IDS module, you must have the module attached to the CTOS workstation, and IDMSS software installed on it.*

1. Type *IMRun* and press the RETURN key.
2. Enter the information in the form that appears

```
IMRun
  Processor ID  #
  Run File      [Sys]<Sys>BmSRF.Run
  [Case]
  [Command]
  [Parameter 1] n
  [Parameter 2]
 (etc...)
```

The value # is the processor ID of the IDS or enhanced IDS module on which you are installing the Suspend/Resume system service. The run file, *[Sys]<Sys>BmSRF.Run*, is the run file for the Suspend/Resume system service. The parameter *n* is some integer number from 1 to 128; this number represents the configured maximum number of simultaneous Suspend/Resume sessions. You must enter a valid number for this parameter or installation of the Suspend/Resume system service will fail.

3. When you are finished entering the information, press the GO key.

Automatic Installation on a CTOS Workstation at Bootup

To install the Suspend/Resume filter on a CTOS workstation so that it is available at bootup, edit the file *[Sys]<Sys>SysInit.Jcl*. On the line immediately following the line entry that installs the BMULTI gateway, enter the line:

```
$RUN [Sys]<Sys>BmSRF.Run,n
```

The parameter *n* is some integer number from 1 to 128; this number represents the configured maximum number of simultaneous Suspend/Resume sessions. You must enter a valid number for this parameter or installation of the Suspend/Resume system service will fail.

Automatic Installation on the IDS Module at Bootup

To install the Suspend/Resume filter on an IDS or enhanced IDS module attached to a CTOS workstation so that the filter is available at bootup, edit the file *[Sys]<Sys>SysInit.Jcl*. Enter the line:

```
$IMRUN #,[Sys]<Sys>BmSRF.Run,,,,n
```

The parameter # is the processor ID of the IDS module. The parameter *n* is some integer number from 1 to 128; this number represents the configured maximum number of simultaneous Suspend/Resume sessions. You must enter a valid number for this parameter or installation of the Suspend/Resume system service will fail.

Automatic Installation on the Shared Resource Processor at Bootup

Note: There is no manual method of installing the BMULTI Suspend/Resume filter as a system service on a shared resource processor.

To install the Suspend/Resume filter on a shared resource processor so that it is available at bootup, edit the file system initialization JCL batch file.

On the XE520, this file is either [Sys]<Sys>InitCP*xx*.Jcl or [Sys]<Sys>InitTP*xx*.Jcl for cluster and terminal processor boards, respectively. The number *xx* represents the board number on your system.

On the XE-530, this file is called [Sys]<Sys>SRPSysInit.Jcl. This file is divided into sections, one for each of the cluster or terminal processor boards used.

In the appropriate JCL file, on the line immediately following the line entry that installs the BMULTI gateway, enter the line:

```
$RUN [Sys]<Sys>BmSRF.Run,n
```

The parameter *n* is some integer number from 1 to 128; this number represents the configured maximum number of simultaneous Suspend/Resume sessions. You must enter a valid number for this parameter or installation of the Suspend/Resume system service will fail.

Deinstalling the Suspend/Resume Filter

The **Deinstall BMULTI S/R Filter** command removes the Suspend/Resume filter from background processing. This allows you to change installation parameters and reinstall the system service without having to reboot.

- When loaded on a CTOS workstation, you must deinstall the Suspend/Resume filter on the workstation on which you installed it. You cannot deinstall the filter from a remote node.
- You must finish any application using the filter before using the deinstall utility.
- The deinstall utility must be executed in the primary partition of the workstation in which the filter is running. You cannot, therefore, deinstall the filter when running Context Manager.
- You cannot deinstall the filter from a shared resource processor or the IDS module.

To execute the deinstall utility from the Executive, type *Deinstall BMULTI S/R Filter* on the Executive command line and press the GO key.

How the Suspend/Resume Filter Works

When installed as a system service for BMULTI, the Suspend/Resume filter operates by intercepting certain BMULTI Low–Level Interface (LLI) procedural calls. Refer to Figure 12–1 for a diagram showing how the filter works with the BMULTI gateway and BMULTI applications.

Two LLI calls, BmResume and BmSuspend, are used specifically with the Suspend/Resume filter and in fact cannot be used unless the filter system service has been installed. The filter also intercepts any BmCommand [End Session] requests. The filter ignores all other BMULTI requests.

Note: *When using BMULTI/X.25, the X.25 virtual circuit is established when the BmCommand [Online] is issued and cleared when BmCommand [Offline] is issued. Because of this, the application must be in the Idle state, and not the Offline state, for Suspend/Resume operations.*

Sessions that do not use the Suspend/Resume filter are not in any way affected by its functioning.

Initiating a Suspend/Resume Session

When an application initiates a session, it must signal to the filter that it wants the session to be a Suspend/Resume session. Therefore, the application must use the BmResume call instead of the BmOpenII call to initiate the session. If the application uses BmOpenII instead of BmResume, the session is not registered with the filter and Suspend/Resume services are not provided.

When an application makes a BmResume call, the filter receives the request. It first checks to see whether there is already a session in progress at the station address indicated in the BmResume request.

If there is no session in progress, the Suspend/Resume filter will establish a session. If the address is already being used by a non–Suspend/Resume application, the filter passes back the BMULTI error code 32772, *Invalid Address*, to the application.

Assuming the connection is successful, the filter records the value of the station handle returned by the gateway, and passes that value back to the application that originally issued the BmResume request. From this point on, the application controls the session normally.

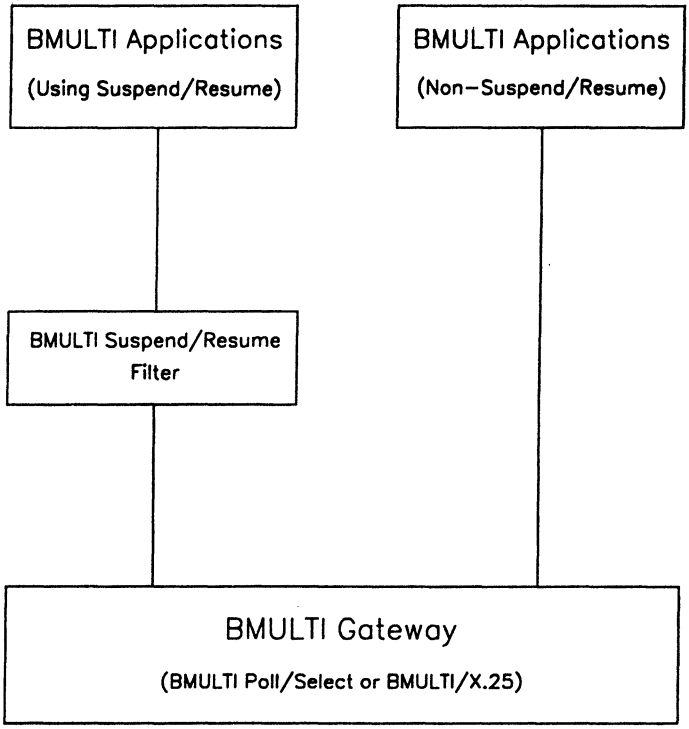

Figure 12–1. The Suspend/Resume Filter

Suspending the Host Session

To suspend an active session, the application issues a BmSuspend request; when suspending a session, the application should be in the Idle state. If successful, the session remains active, but is considered to be in the suspended state. Any application issuing a BmResume request for the address associated with that session can then resume control.

If the BmSuspend request cannot be accepted, for example, when the application state is "Receive Ready," "Receiving," or "Transmitting," the Suspend/Resume filter responds back to the application with error code 60260, *Application Cannot Be Suspended.*

If a non–Suspend/Resume application—one that did not initiate the original connection by using BmResume instead of BmOpenII—sends a BmSuspend call, the filter responds with error code 60264, *Invalid Command*. This error code is also received if the application attempts to suspend a session that does not exist.

If the filter receives a BmResume call, and all configured Suspend/Resume sessions are in use, the filter responds with error code 60262, *Suspend/Resume session overflow*.

Resuming the Host Session

To resume a suspended session, the application issues a BmResume request (in the same way it initiates a session supporting Suspend/Resume capabilties). If the session for the indicated BMULTI address is in the suspended state, the filter passes the station handle back to the requesting application. The application then uses the station handle for session communications.

If some other Suspend/Resume–registered application is currently using the requested address, the filter responds to the application with error code 60261, *Session Active*.

If the address is already being used by a non–Suspend/Resume application, BMULTI responds with error code 32772, *Invalid Address*, and this error code is passed by the filter back to the application.

Ending a Suspend/Resume Host Session

To end a session gracefully, an application must issue a BmCommand [End Session] request; this should be done in the Idle state. This will close the session. If another application attempts to resume the same address, a new session will be established.

If the BmCommand [End Session] is issued by an application that is in the suspended state, the session is not terminated.

Using the Suspend/Resume Filter with MGS

Although the BMULTI Suspend/Resume filter supports the use of CTOS Multiple Gateways Service (MGS), multiple copies of the filter cannot be installed. Instead, the filter is installed outside MGS. Refer to Figure 12-2 for a diagram showing the installation of the Suspend/Resume filter with MGS.

Specifically, the BMULTI gateways (including BMULTI/X.25, if present) are installed under MGS. This is done by first installing MGS as a system service, then by installing the BMULTI gateways using the Service ID parameter. Finally, you install a single instance of the BMULTI Suspend/Resume filter.

All application procedural call requests for the Suspend/Resume filter are intercepted by the filter and routed for each session based on the Service ID and station address.

When multiple copies of BMULTI gateways are installed, the maximum number of configurable Suspend/Resume sessions is 128. The same poll/select address can be valid for each of the gateways.

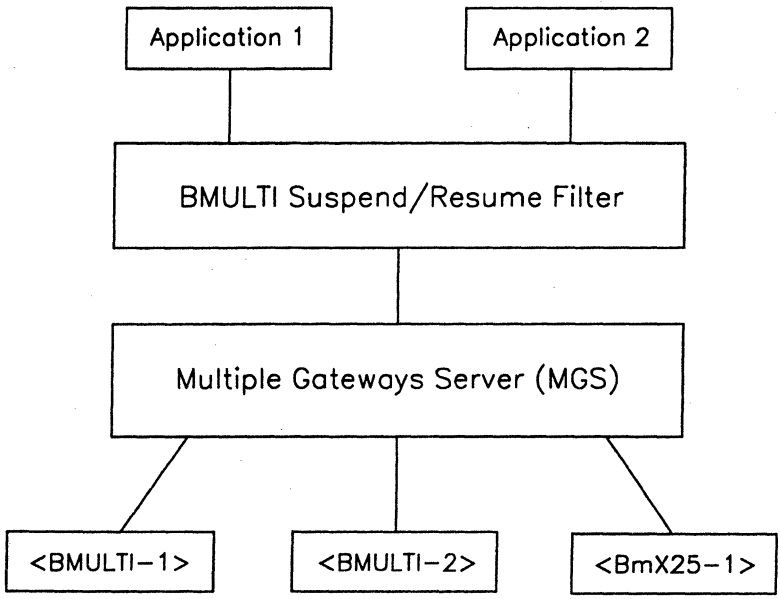

Figure 12–2. The Suspend/Resume Filter with MGS

Appendix A
Status Codes and Troubleshooting

Status Codes

Status Code (Decimal)	Explanation
60	BMULTI cannot be installed on a DCX module because the DCX server is not installed.
65	NRZI encoding, TDI, or X.21 electrical interface is not supported by the hardware in the specified channel.
8401	Line in use. This code may also indicate that BMULTI attempted to use X.21 on a channel that does not support X.21.
31309	Invalid or duplicate Service ID.
31310	Too many gateway types installed under the Multiple Gateways Service (MGS).
31311	Too many copies of BMULTI installed under the Multiple Gateways Service (MGS).
31312	Too many requests issued in using BMULTI under the Multiple Gateways Service (MGS).
31600	DMA I/O is not supported by the hardware on the specified channel.
32768	An invalid command was issued to BMULTI.
32769	Task overflow. The multitasking interface cannot handle more than three device addresses.
32770	Command pending. Only one BMULTI command can be outstanding at any time per address.

Status Code (Decimal)	Explanation
32771	Report pending. Only one BMULTI report request can be outstanding at any time for an address.
32772	Invalid BMULTI address.
32773	Command denied (Mp Interface).
32774	Buffer overflow.
32775	Invalid report request.
32776	Read in progress.
32777	Write in progress.
32778	A request for a large buffer transfer cannot be honored at this time due to a lack of buffer space. Try again later or reconfigure with a larger buffer size.
32779	Invalid buffer length for this command.
32780	Offline command denied.
32781	Online command denied.
32782	Idle command denied.
32783	Fast Ready command denied.
32784	Transmit Ready command denied.
32785	Receive Ready command denied.
32786	Transfer Transmit Buffer command denied.
32787	Transfer Receive Buffer command denied.
32788	End Session command denied.
32789	Station not online.
32790	Station overflow.
32791	Device address is identical to group address.
32792	The buffer received by the application is only part of the full message.

Status Code (Decimal)	Explanation
32793	Internal error (report it to the system administrator).
32794	Duplicate virtual address or duplicate X.21 poll/select addresses are configured when attempting to configure BMULTI for X.21 circuit switched mode.
32795	BMULTI locked for reconfiguration.
32796	Entry error or no poll/select addresses are configured when attempting to configure BMULTI for X.21 circuit switched mode.
32797	Station active. Context Manager cannot swap tasks at this time.
32798	BMULTI was not deinstalled because the operating system is single partition.
32799	BMULTI was not deinstalled, either because the deinstall utility was not executed in the primary application partition (Context Manager installed) or because the Deinstall utility was not executed on the machine on which BMULTI is installed.
32800	Line monitor facility is in use.
32801	Data lost while monitoring.
32802	BMULTI was not deinstalled because stations are active.
32803	An attempt to call BmOpenII, requesting an unused virtual address, failed because there are no more unused virtual addresses.
32804	Inconsistency between BMULTI and BmZip, BmConfig, or BmStatus. If you have configuration files created with a previous release level of BMULTI, you must replace them with new configuration files from the current release level (see "Configuring and Installing BMULTI" in Section 4).

Status Code (Decimal)	Explanation
32805	A change of extended character translation mode was attempted by BmSetXlatMode or SetXlatModeBMULTI when station was not Idle, or the station is Online but not Fast Ready.
32806	An invalid iRcvMode or iXmitMode was specified for BmSetXlatMode or SetXlatModeBMULTI.
32807	The character translation file is incomplete or in unexpected format.
32808	The data contents of the character translation file are not valid ASCII. Valid characters are 0–9, A–F, a–f, or space (30h–39h, 41h–46h, 61h–66h, or 20h).
32809	BmGetStatus (iRecordLen * nStation) exceeds specified buffer size.
32810	The number of stations or IO buffer size cannot be reconfigured dynamically.
32811	BMULTI has already been installed.
32812	An invalid node specification was passed to SelectBMULTI. See the "SelectBMULTI" procedural interface in Section 10.
32813	The report queue of the station requesting the report has overflowed and the contents of the queue are now meaningless.
56252	Multiple Gateways Service (MGS) is not available. You attempted to install BMULTI using a Service ID without first installing the MGS server.
56253	Installation failed because a copy of BMULTI is already installed outside MGS.
56254	Active poll/select address not specified during dynamic reconfiguration.
56255	DMA initialization attempted after BMULTI was already successfully installed without DMA.
56256	Host connection cannot be established.

Status Code (Decimal)	Explanation
60260	Application cannot be suspended.
60261	Session active. An active session must be suspended before it can be resumed by another application.
60262	Suspend/Resume session overflow. All available configured Suspend/Resume sessions have been used up.
60263	VADs not supported. Virtual addresses cannot be used as Suspend/Resume addresses.
60264	Invalid command. An application attempted to issue a suspend request on an active connection that was not initiated through the Suspend/Resume filter.
60265	Invalid Service ID. A BmResume call was issued with an invalid Service ID in the parameter fields.
60266	Application suspended. The current session is in the suspended state and will refuse all commands until the session is resumed.
60267	Cancel command denied. The application state is "Receive Ready," "Receiving," or "Transmitting," and the session cannot be suspended.

Troubleshooting Notes

Receive Buffer Overflow

When writing BMULTI applications, you are responsible for creating a receive buffer of adequate size. If the buffer does overflow, that is, the ASM RcvStatus field equals 14, you should rewrite your application to issue a delay of approximately 1 second or issue a ResetBMULTI (HLI Interface command) prior to issuing another ReadBMULTI.

Transmission Blocks

Applications can require a maximum of three transmission blocks per address. The number of requests an application usually will have outstanding is unchanged, but the new maximum is three. CTOS Serv operating systems may not operate optimally if the number of XBLKs (YBLKs and ZBLKs for the XE 520) is not at least three times the numb of BMULTI addresses to be used. On shared resource processors, these are divided proportionally between the cluster processors.

Too few XBLKs results in poor performance. Occasionally, a deadlock situation can arise. In this situation, all existing XBLKS hold requests that cannot be responded to unless more requests are made. More requests cannot be made, however, because all the XBLKs are in use.

Another problem caused by having too few XBLKs on the server system occurs when client stations cannot issue Abort requests to return Transmitting to the Idle state (that is, when using BTE, the operator presses the Local key while the F10 LED is lit).

BNet Support

The High–Level Interface and Low–Level Interface support routing over BNet. A properly written application linked with the HLI or LLI can access a copy of BMULTI that is running on a remote BNet node.

If you are running across BNet, the BNet installation parameter values may need to be changed. In the Net Agent installation form, there is a parameter:

[Max # client requests queued (default=8)]

In the Net Server installation form, there is a parameter:

[Max # data buffers (default=4)]

The values used for both forms should be at least three times the numbe of BMULTI address in use, plus two. This is similar to the transmission block XBLK/YBLK calculations, but applies only to BMULTI addresses being used across BNet.

BMOpenII, Virtual Addresses, and Existing Applications

Applications that call BmOpenII with the "any open virtual address" option must be introduced carefully into existing installations, especially if they are being mixed with older applications with fixed addresses. The new application may request and be granted an unused virtual address that is customarily used by another application, with the result being that you cannot install the other application.

Virtual Addresses and Network Definition Languages

Virtual addresses (VADs), which are optional, should be used only with unsophisticated Network Definition Languages (NDLs). Acceptable NDLs are typically ones used on CMS machines and Unisys small systems. The NDLs used in these environments send poll sequences to inactive stations until a timeout or the number of retries is exceeded. After either of these conditions becomes true, that station is removed from the poll sequence and can be reactivated only through the host operator's intervention. This is the type of environment in which VADs are particularly useful.

Indiscriminately using VADs on systems with sophisticated NDLs can cause a noticeable performance degradation on the data communications line. This is more noticeable when a large number of virtual addresses are defined in the BMULTI configuration file.

The NDLs used on large systems (for example, the B7900) and the A Series mainframes preclude the use of virtual addresses. NDLs on these systems poll inactive addresses less frequently than active addresses, so that the line is not overloaded with unnecessary polls and selects. Do not use virtual addresses in these environments; if a station becomes active, the host eventually starts polling that terminal without any additional operator intervention.

Host/NDL Timeout Values

In some cases, BMULTI may be inherently slower than a terminal, because terminals have processors dedicated to data communications processing. Host timeout and transmit delay parameters, therefore, must be set to higher values when running networks that include BMULTI stations.

You should use a minimum host timeout value for server/client configurations of 1000 milliseconds (1 second). Use a 25 to 50 millisecond range for the transmit delay.

You may need to modify these settings depending upon the actual performance of the network. In networks where all applications are linked with BMULTI 6.0 or higher, host timeout values can be set as low as 50 milliseconds. Applications must use the High–Level or Low–Level interface to take advantage of the performance improvements of the 6.0 and higher libraries.

Configuration Parameters

If BMULTI does not operate, make sure the values for baud rate or time delay are correct for the host you are operating as well as for BMULTI. Although you may have to enter values that are accepted by BMULTI, they may not be correct for your current host system.

Response Time

BMULTI allows up to 64 addresses to be used by applications at one time. Response time, however, varies according to the number of addresses in use, as well as with the amount of data traffic generated by the applications.

For example, you may be satisfied with the response time if you are running ten copies of a low–traffic, three–address application. Ten copies of a high–traffic, multi–address application, though, may result in an unsatisfactory response time for your requirements and applications.

For these situations, you may want to consider shifting some BMULTI operations to other workstations, perhaps through BNet. Another alternative would be to rewrite your applications to reduce the data communications requirements.

Configurable Delays

The configurable delays offered by BMULTI are upwardly variable because some processes in the operating system have a higher priority than BMULTI. The actual delay is never less than that with which BMULTI is configured, but it could be higher than configured due to workstation load and other considerations.

Choosing the Channel to Use with the DCX Module

You must install the DCX hardware and server software before you install BMULTI if you have configured BMULTI to use channels 1A, 1B, 1C, 1D, 2A, 2B, 2C, or 2D. If this is not done, BMULTI will fail to install.

Status Code 32786, "Transfer Xmt Buffer Command Denied"

Status code 32786 is displayed occasionally when the Transfer Transmit Buffer command is issued. This usually happens because the application did not check the report queue often enough, causing it to overflow. To avoid this problem, have the application empty the queue before issuing the SetTransmitReady command.

Put as little code as possible between checking for a report of the Ready for Transmit buffer and issuing the commmand Transfer Transmit Buffer. Also, put as little code as possible between CTOS calls.

To recover from this problem, issue an Idle command and reinitiate the transmit sequence.

Appendix B
USASCII Code Charts

b7 b6 b5 →					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
$B_{I}T_{S}$ b4 b3 b2 b1				ROW ↓ COLUMN →	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE		0				POL
0	0	0	1	1	SOH	DC1		1				SEL
0	0	1	0	2	STX	DC2						
0	0	1	1	3	ETX	DC3						FSL
0	1	0	0	4	EOT	DC4						BSL
0	1	0	1	5	ENQ	NAK						
0	1	1	0	6	ACK	SYN						
0	1	1	1	7	BEL*	ETB						
1	0	0	0	8	BS	CAN						
1	0	0	1	9	HT	EM						
1	0	1	0	10	LF	SUB						
1	0	1	1	11	VT	ESC				[		
1	1	0	0	12	FF	FS		<				
1	1	0	1	13	CR	GS				]		
1	1	1	0	14	SO	RS						
1	1	1	1	15	SI	US						DEL

* CON (ALTERNATE CODE FOR CONTENTION)

Figure B–1. Universal Control Codes and Special Allocation of Codes to Implement with BMULTI Protocol

b7 b6 b5 →					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
BITS	b4	b3	b2	b1	ROW ↓ COLUMN →	0	1	2	3	4	5	6	7
	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
	0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
	0	0	1	0	2	STX	DC2	"	2	B	R	b	r
	0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
	0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
	0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
	0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
	0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
	1	0	0	0	8	BS	CAN	(	8	H	X	h	x
	1	0	0	1	9	HT	EM	)	9	I	Y	i	y
	1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
	1	0	1	1	11	VT	ESC	+	;	K	[	k	{
	1	1	0	0	12	FF	FS	,	<	L	\	l	|
	1	1	0	1	13	CR	GS	-	=	M	]	m	}
	1	1	1	0	14	SO	RS	.	>	N	^	n	~
	1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Figure B–2. USA Standard Code for Information Interchange (USASCII)

Appendix C
CTOS Request Codes for BMULTI

BMULTI uses the following request codes.

BMULTI CTOS Requests

Request Code (hex)	Function
7165h	Swap request
7166h	Change user number request
7167h	Abort request
7168h	Terminate request

BMULTI Client Requests

Request Code (hex)	Function
7137h	Utility request
7169h	Deinstall request
716Ah	Line Monitor Query request
73FFh	Deinstall S/R filter request
7400h	Suspend session request
7401h	Resume session request
C0D0h	Get Translate Table request
C0D2h	Get Status request
D02Ah	Report request
D02Bh	Open request
D02Dh	Command request
D02Eh	End Session request

Appendix D
Examples of Shift–In and Shift–Out Action

The following examples clarify the actions occurring in the transmit and receive modes selected by the BmSetXlatMode and SetXlatModeBMULTI procedures.

CTOS Receiving Format

Original Data Stream	Data As Received By BTOS		Resultant User Buffer		
Position	ASCII	Hex	Mode 0	Mode 1	Mode 2
1	A	41	41	41	41
2	ESC	1B	1B		
3	SO	0E	0E		
4	B	42	42	C2	C2
5	ESC	1B	1B	9B	9B
6	3	33	33	B3	B3
7	1	31	31	B1	B1
8	ESC	1B	1B	9B	9B
9	'	27	27	A7	A7
10	.	2E	2E	AE	AE
11	ESC	1B	1B		
12	SI	0F	0F		
13	C	43	43	43	43
14	D	44	44	44	44
15	SO	0E	0E	0E	
16	E	45	45	45	C5
17	SI	0F	0F	0F	
18	3	33	33	33	33

Note: *In Mode 1, ESC–SO and ESC–SI are not buffered. In Mode 2, ESC, ESC–SO, ESC–SI, SO, and SI are not buffered.*

CTOS Transmitting Format

Original Data Stream	Data As Received By BTOS		Resultant User Buffer		
Position	ASCII	Hex	Mode 0	Mode 1	Mode 2
1	a	61	61	61	61
2	<space>	20	20	20	20
				1B	ESC
				0E	SO
3	n	EA	6A	6A	6A
				1B	ESC
				0F	SI
4	f	66	66	66	66
5	l	6C	6C	6C	6C
6	u	7D	7D	7D	7D
7	v	7E	7E	7E	7E
8	e	65	65	65	65
9	1	31	31	31	31
10	2	32	32	32	32
				1B	
				0E	0E
11	a	D3	53	53	53
12	e	DF	5F	5F	5F

Appendix E
Translation Tables

The character sets for various languages supported by the CTOS systems are not the same as those on Unisys mainframes. This is true for standard characters (00h–7Fh) as well as for extended characters (in the 80h–FEh range). As a result, some translation between character sets is necessary, both before transmission and when received. To accomplish this, BMULTI uses a translation table that is loaded from a disk file (specified in the BMULTI configuration file). The default translation file is called BMULTIXlat.sys, and is the USA version.

BMULTI performs the translation of the corporate standard character set to the local CTOS character set, and vice versa. Characters that appear in only one of the character sets are translated to be a suitable character when possible, otherwise a question mark is substituted. Characters in conflict with the local CTOS character set are remapped above 80h. The template in this appendix shows the mapping for the seven different languages generally supported by the CTOS terminal emulators.

There are really two character translation tables built into the specified translation file. The first table is used to remap 256 characters from the mainframe to the CTOS workstation, and the other is used to remap 256 characters from the CTOS workstation to the host. You should never remap the ETX character (03h) to any other value, or remap any other character to be an ETX. In general, it is not advisable to remap any poll/select protocol characters. BMULTI does not apply the translation tables to its own protocol.

BMULTI can use only a single translation file at any given time, and it is shared by all users of that copy of BMULTI. BMULTI loads this translation table file (specified in the configuration file) from disk when it is installed, but the table can be changed dynamically by specifying a new translation table file using the Reconfigure BMULTI command. You should be aware that this will change the translation table for all applications. Any translation table file to be used by BMULTI must reside on disk.

The format of a translation table file on disk is that of an editable text file. These table files are a matrix of hex character values separated by spaces, as shown in the supplied template. Comments can also be entered in this file. Comments and data alternate between pairs of colon delimiters. The file is always assumed to begin with a comment, and everything prior to the first colon is ignored. When BMULTI is installed or reconfigured, the tables are loaded into RAM. When this happens, comments and delimiters are removed, and the two byte ASCII data entries are packed into a single byte format (one hexadecimal byte per character). This results in a 256 byte transmit translation table and a 256 byte receive translation table. The 3Fh character in the template is a question mark, used when no reasonable cross–mapping of characters exists.

If there is an odd number of data bytes (anything other than spaces) between colons, or if the end of file is encountered before the construction of the tables is complete, an InvalidFileFormat error is returned. If data field characters outside the ranges of 30h–39h (0–9) inclusive or 41h–46h (A–F) inclusive are encountered, an InvalidFileData error is returned.

Language Version 01: United States

Receive Translation Table E–1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	3F	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	3F	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	3F	3F	3F	3F	3F	3F
3:	03	13	23	33	43	53	63	73	3F	3F	3F	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	3F	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	3F	3F	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	3F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	3F	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	3F	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	3F	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	3F	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	3F	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	3F	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	3F	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–2

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	41	42	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	61	3F	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	41	3F	3F	3F	3F	3F
3:	03	13	23	33	43	53	63	73	3F	3F	61	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	4F	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	6F	3F	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	4F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	6F	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	55	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	75	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	63	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	65	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	65	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	65	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

Language Version 02: United Kingdom

Receive Translation Table E–3

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	3F	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	3F	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	3F	3F	3F	3F	3F	3F
3:	03	13	B0	33	43	53	63	73	3F	3F	3F	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	3F	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	3F	3F	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	3F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	3F	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	3F	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	3F	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	3F	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	3F	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	3F	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	3F	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–4

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	23	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	7C	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	3F	60	3F	3F	3F	3F
3:	03	13	3F	33	43	53	63	73	3F	3F	61	27	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	61	5E	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	61	3F	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	63	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	65	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	65	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	65	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	69	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	6F	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	6F	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	75	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	75	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	75	3F	3F	3F	3F	3F

Language Version 03: France

Receive Translation Table E–5

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	B1	41	44	A0	61
1:	01	11	21	31	41	51	61	71	3F	3F	3F	3F	41	4E	61	6E
2:	02	12	22	32	42	52	62	72	3F	3F	3F	3F	41	4F	A1	6F
3:	03	13	23	33	43	53	63	73	3F	3F	B2	3F	41	4F	61	6F
4:	04	14	24	34	44	54	64	74	3F	3F	3F	3F	41	4F	A2	AA
5:	05	15	25	35	45	55	65	75	3F	3F	3F	3F	41	4F	61	6F
6:	06	16	26	36	46	56	66	76	3F	3F	7C	3F	41	4F	3F	AB
7:	07	17	27	37	47	57	67	77	3F	3F	B0	3F	3F	3F	A3	3F
8:	08	18	28	38	48	58	68	78	3F	3F	B4	3F	43	45	A5	6F
9:	09	19	29	39	49	59	69	79	3F	3F	3F	3F	45	55	A4	AC
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	3F	3F	45	55	A6	75
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	3F	3F	45	55	A7	AD
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	3F	3F	45	55	69	AE
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	3F	3F	49	59	69	79
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	49	70	A8	70
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	2D	3F	49	42	A9	79

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–6

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	E0	A7	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	E2	B0	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	E4	A3	3F	3F	3F	3F
3:	03	13	23	33	43	53	63	73	3F	3F	E7	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	E9	A8	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	E8	5E	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	EA	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	EB	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	EE	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	EF	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	F4	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	F6	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	F9	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	FB	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	FC	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

Language Version 04: Netherlands

Receive Translation Table E–7

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	3F	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	3F	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	3F	3F	3F	3F	3F	3F
3:	03	13	A4	33	43	53	63	73	3F	3F	3F	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	3F	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	3F	3F	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	3F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	3F	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	3F	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	3F	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	3F	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	3F	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	3F	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	3F	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–8

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	3F	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	5E	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	3F	3F	3F	3F	3F	3F
3:	03	13	3F	33	43	53	63	73	3F	3F	66	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	23	60	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	3F	27	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	3F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	61	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	65	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	65	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	6F	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	75	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	61	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	65	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	75	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	65	3F	3F	3F	3F	3F

Language Version 05: Germany

Receive Translation Table E–9

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	B5	50	60	70	3F	3F	3F	45	A1	6E	65	A4
1:	01	11	21	31	41	51	61	71	3F	3F	41	47	4F	A7	67	6F
2:	02	12	22	32	42	52	62	72	3F	3F	41	49	53	61	69	73
3:	03	13	23	33	43	53	63	73	3F	3F	41	49	53	AB	69	73
4:	04	14	24	34	44	54	64	74	3F	3F	41	49	3F	61	AD	AB
5:	05	15	25	35	45	55	65	75	3F	3F	A0	49	50	A3	69	70
6:	06	16	26	36	46	56	66	76	3F	3F	41	49	55	61	69	A9
7:	07	17	27	37	47	57	67	77	3F	3F	3F	3F	55	3F	3F	75
8:	08	18	28	38	48	58	68	78	3F	3F	43	4A	55	63	6A	AF
9:	09	19	29	39	49	59	69	79	3F	3F	43	4C	A2	63	6C	A5
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	43	4E	57	63	63	77
B:	0B	1B	2B	3B	4B	A0	6B	A3	3F	3F	44	3F	59	3F	3F	79
C:	0C	1C	2C	3C	4C	A1	6C	A4	3F	3F	3F	4F	59	64	6F	79
D:	0D	1D	2D	3D	4D	A2	6D	A5	3F	3F	45	4F	59	A8	6F	79
E:	0E	1E	2E	3E	4E	5E	6E	A6	3F	3F	45	4F	5A	AA	AE	7A
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	45	4F	3F	AC	6F	3F

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–10

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	3F	50	60	70	3F	3F	5B	60	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	5C	27	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	5D	5E	3F	3F	3F	3F
3:	03	13	23	33	43	53	63	73	3F	3F	7B	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	7C	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	7D	40	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	7E	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	D1	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	DD	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	F6	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	DE	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	3F	6B	3F	3F	3F	D3	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	3F	6C	3F	3F	3F	DF	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	3F	6D	3F	3F	3F	E4	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	3F	3F	3F	EE	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	F8	3F	3F	3F	3F	3F

Language Version 06: South Africa

Receive Translation Table E–11

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	3F	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	3F	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	3F	3F	3F	3F	3F	3F
3:	03	13	B0	33	43	53	63	73	3F	3F	3F	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	3F	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	3F	3F	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	3F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	3F	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	3F	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	3F	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	3F	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	3F	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	3F	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	3F	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–12

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	5E	23	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	3F	3F	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	27	3F	3F	3F	3F	3F
3:	03	13	3F	33	43	53	63	73	3F	3F	60	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	65	3F	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	65	6E	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	6F	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	6F	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	61	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	65	3F	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	65	3F	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	75	3F	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	61	3F	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	75	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	3F	3F	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	3F	3F	3F	3F	3F	3F

Language Version 07: Canada

Receive Translation Table E–13

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	3F	BB	4F	6E	AA	6F
1:	01	11	21	31	41	51	61	71	3F	3F	41	47	4F	A0	3F	6F
2:	02	12	22	32	42	52	62	72	3F	3F	41	49	53	61	3F	73
3:	03	13	23	33	43	53	63	73	3F	3F	B4	49	53	A1	3F	73
4:	04	14	24	34	44	54	64	74	3F	3F	41	B0	3F	61	A6	42
5:	05	15	25	35	45	55	65	75	3F	3F	41	49	50	61	3F	70
6:	06	16	26	36	46	56	66	76	3F	3F	41	BC	AF	61	AB	A8
7:	07	17	27	37	47	57	67	77	3F	3F	3F	3F	55	3F	3F	75
8:	08	18	28	38	48	58	68	78	3F	3F	43	4A	B9	63	3F	A9
9:	09	19	29	39	49	59	69	79	3F	3F	B2	4C	BE	A2	3F	AC
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	43	4E	57	63	3F	77
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	44	3F	59	3F	3F	79
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	3F	4F	59	3F	3F	79
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	AE	4F	59	A4	3F	79
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	BA	B1	5A	A3	A7	7A
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	B5	4F	3F	A5	3F	3F

The numbers and letters along the top row indicate column numbers; those along the lefthand column indicate row numbers. 3F is a question mark.

Transmit Translation Table E–14

Extended character translations are made to the closest related character (for example, an "a umlaut" is translated to a plain "a"). A question mark (3Fh) is used when there is no suitable character.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	00	10	20	30	40	50	60	70	3F	3F	D1	B4	3F	3F	3F	3F
1:	01	11	21	31	41	51	61	71	3F	3F	D3	BE	3F	3F	3F	3F
2:	02	12	22	32	42	52	62	72	3F	3F	D9	A9	3F	3F	3F	3F
3:	03	13	23	33	43	53	63	73	3F	3F	DE	3F	3F	3F	3F	3F
4:	04	14	24	34	44	54	64	74	3F	3F	DD	A3	3F	3F	3F	3F
5:	05	15	25	35	45	55	65	75	3F	3F	DF	AF	3F	3F	3F	3F
6:	06	16	26	36	46	56	66	76	3F	3F	E4	3F	3F	3F	3F	3F
7:	07	17	27	37	47	57	67	77	3F	3F	EE	3F	3F	3F	3F	3F
8:	08	18	28	38	48	58	68	78	3F	3F	F6	3F	3F	3F	3F	3F
9:	09	19	29	39	49	59	69	79	3F	3F	F8	C8	3F	3F	3F	3F
A:	0A	1A	2A	3A	4A	5A	6A	7A	3F	3F	E0	AE	3F	3F	3F	3F
B:	0B	1B	2B	3B	4B	5B	6B	7B	3F	3F	E6	B0	3F	3F	3F	3F
C:	0C	1C	2C	3C	4C	5C	6C	7C	3F	3F	F9	B6	3F	3F	3F	3F
D:	0D	1D	2D	3D	4D	5D	6D	7D	3F	3F	A1	3F	3F	3F	3F	3F
E:	0E	1E	2E	3E	4E	5E	6E	7E	3F	3F	AD	C9	3F	3F	3F	3F
F:	0F	1F	2F	3F	4F	5F	6F	7F	3F	3F	C6	3F	3F	3F	3F	3F

Appendix F
BMULTI Configuration Worksheet

You can make photocopies of the configuration worksheet in this appendix as an aid in configuring BMULTI. In particular, the worksheet can be a convenient way to supply operators with BMULTI configuration parameters. Also, completed worksheets can be saved or archived against the possible future need to reconfigure BMULTI.

Anyone familiar with CTOS workstations can use the BMULTI Configuration File Editor (described in Section 4) to enter the values from this worksheet. For instructions on how to configure BMULTI and definitions of the configuration parameters, refer to Section 4, "Configuring and Installing BMULTI."

If you need to report a problem, attach a photocopy of the worksheet with your current configuration parameters to a User Communication Form (UCF) and send it to the appropriate Resource Control Center or Customer Support Center. The UCF should be filled out completely, including information about your specific hardware and software environment.

Translation table name
(default = *[Sys]<Sys>BMULTIXlat.Sys*) _____

X.21 address file name (default = none) _____

Modem phone number (default = none) _____

Screen Message	Default Value	Possible Values	Change to
Group Poll Address	dv	020h–07Fh	
Group Select Character	r	020h–07Fh	
Synch or Asynch	A	'A' or 'S'	
Channel	0B	A through H, 0A, 0B, 0C, 0D, 1A, 1B, 1C, 1D, 2A, 2B, 2C, 1D	
Bit Rate	09600	110–38400	
Transmission Numbering	0	0, 1, 2, 3, 4, 5	
RTS–CTS delay	0	0–255	
XMT–RCV delay	0	0–255	
RTS HOLD delay	0	0–255	
Anything Downstream?	N	Y or N	
TDI on IDM?	N	Y or N	
Number of Stations	16	1–64	
Buffer Size	2048	16–4096	
Service ID	none	*<1–12 chars>*	
DMA?	N	Y or N	
X.21	D	D, L, S	
Line Inactivity Time	0	0–999	
Control Switching	N	Y or N	
Call Hold Time	0	0–999	
Call Direction	B	I, O, B	
X.21 CSS Service ID	none	*<1–12 chars>*	
Incoming Address Verification?	N	Y or N	
Dial/Answer/Call Back/Off	O	D, A, C, O	
Call Timeout	0	0–999	
Call Retries	0	0–999	

Appendix G
Sample Programs

This appendix contains sample programs written in Pascal. For the two Pascal programs, the first uses the High–Level Interface (HLI) and the second uses the Low–Level Interface (LLI).

Applications written in any language will not operate unless BMULTI.lib is linked with the executable code file. Applications written in FORTRAN, BASIC, or COBOL will not operate unless the languages have been reconfigured. Refer to Section 7 for information about language configuration.

Note: *The sample programs in this appendix are for demonstration purposes only, showing the general use of BMULTI procedure calls. While the programs may function in some conditions, they may not compile, link, or operate properly in others. For this reason, the sample programs are not considered part of the Unisys–supported BMULTI software product.*

Pascal Terminal Program (Using High–Level Interface)

This program uses the High–Level Interface (HLI) to implement a simple terminal. Keyboard and screen code is shown, as well as data communications handling.

```
{$DEBUG-}
{$ENTRY-}

*PROGRAM  HLITerm;

TYPE
    String2  = STRING(2);
    pbType   = ADS OF BYTE;
    pwType   = ADS OF WORD;
    ppType   = ADS OF pbType;
    psType   = ADS OF String2;

CONST

 (* Miscellaneous*)
    banner           = 'B 20 Mini-Term (HLI)';
    sBanner          = 19;
    ErcOk            = 0;

VAR [PUBLIC]
    Erc              : WORD;
    Report           : WORD;
    th               : WORD;
    DevAdr           : String2;
    cMsg             : INTEGER;
    Msg              : ARRAY[0..79] OF BYTE;
    sBufMax          : INTEGER;
    sBuf             : INTEGER;
    Buff             : ARRAY[0..2047] OF CHAR;
    dummyPtr         : pbType;
    pVidSeg          : pbType;
    sMap             : WORD;
    nLines           : INTEGER;
    Key              : BYTE;
    current_col      : INTEGER;
    vid_col          : INTEGER;
    SdRet            : RECORD
        pSubParam    : psType;
        sSubParam    : WORD;
                       END;
    vHdw             : RECORD
        level        : BYTE;
        nLinesMax    : SINT;
        nColsNar     : BYTE;
        nColsWide    : BYTE;
                       END;
```

```
  ASBlk      : RECORD
    RcvStatus [00] : BYTE;.

    RcvErc    [01] : WORD;
    fSelDen   [03] : BOOLEAN;
    XmtStatus [04] : BYTE;
    XmtErc    [05] : WORD;
    Option    [07] : BYTE;
    fFastMsg  [08] : BOOLEAN;
    pFastMsg  [09] : pbType;
              END;

VALUE
    sBufMax        := 4096;
    sMap           := 16#0B6C;

!#########################################################
!#          System Common Procedures
!#########################################################
PROCEDURE Exit; EXTERN;

PROCEDURE ErrorExit (ercTerm : WORD); EXTERN;

FUNCTION  PosFrameCursor (iFrame : INTEGER;
                          iCol   : INTEGER;
                          iLine  : INTEGER) : WORD; EXTERN;

FUNCTION  PutFrameChars (iFrame : INTEGER;
                         iCol   : INTEGER;
                         iLine  : INTEGER;
                         pbText : pbType;
                         cbText : INTEGER) : WORD; EXTERN;

FUNCTION  ResetFrame ( iFrame : INTEGER) : WORD; EXTERN;

FUNCTION  ScrollFrame ( iFrame     : INTEGER;
                        iLineStart : INTEGER;
                        iLineMax   : INTEGER;
                        cLines     : INTEGER;
                        fUp        : BOOLEAN) : WORD; EXTERN;

!#########################################################
!#          Object Module Procedures
!#########################################################
FUNCTION  CloseBMULTI (Sh : BYTE) : WORD; EXTERN;

FUNCTION  OpenBMULTI (devAdr : String2;
                      fSys   : BOOLEAN;
                      Pri    : WORD;
                      pSh    : pbType;
                      pAsBlk : pbType) : WORD; EXTERN;

FUNCTION  ReadBMULTI (Sh   : BYTE;
                      pBuf : pbType;
                      sBuf : INTEGER) : WORD; EXTERN;

FUNCTION  ResetBMULTI (Sh  : BYTE) : WORD; EXTERN;
```

```
FUNCTION  WriteBMULTI (Sh   : BYTE;
                       pBuf : pbType;
                       sBuf : INTEGER) : WORD; EXTERN;

FUNCTION  RgParam (lParam    : WORD;
                   lSubParam : WORD;
                   pSdRet    : pbType) : WORD; EXTERN;

!##########################################################
!#          Procedural requests
!##########################################################
FUNCTION  InitCharMap (pMap : pbType;
                       sMap : WORD) : WORD; EXTERN;

FUNCTION  InitVidFrame (lFrame      : INTEGER;
                        lColStart   : INTEGER;
                        lLineStart  : INTEGER;
                        nCols       : INTEGER;
                        nLines      : INTEGER;
                        borderDesc  : BYTE;
                        bBorderChar : CHAR;
                        bBorderAttr : BYTE;
                        fDblHigh    : BOOLEAN;
                        fDblWide    : BOOLEAN) : WORD; EXTERN;

FUNCTION  QueryVidHdw (pBuf : pbType;
                       sBuf : WORD) : WORD; EXTERN;

FUNCTION  ReadKbdDirect (mode     : WORD;
                         pCharRet : pbType) : WORD; EXTERN;

FUNCTION  ResetVideo (nCols    : INTEGER;
                      nLines   : INTEGER;
                      fAttr    : BOOLEAN;
                      bSpace   : CHAR;
                      psMapRet : pbType) : WORD; EXTERN;

FUNCTION  SetScreenVidAttr (lAttr : WORD;
                            fOn   : BOOLEAN) : WORD; EXTERN;

!##########################################################
  PROCEDURE Check_Erc (lrk : WORD)  [PUBLIC];
!##########################################################
BEGIN
  IF (lrk <> 0) THEN ErrorExit (lrk);
END;

!##########################################################
  PROCEDURE Screen_setup  [PUBLIC];
!##########################################################
VAR
  BannerStart : INTEGER;.
BEGIN
  Check_Erc (QueryVidHdw (ADS vHdw, 4));

  nLines := vHdw.nLinesMax;
  Check_Erc (ResetVideo (80, nLines, FALSE,
                         ' ', ADS sMap));
```

```
!#########################################################
!##  Frame 0 is the line at the top of the screen.
!##  It has a solid thin border.
!#########################################################
  Check_Erc (InitVidFrame (0, 0, 0, 80, 1,
                           4, CHR(#0DA), 0, FALSE, FALSE));
!#########################################################
!##  Frame 1 is from line 3 to one line above the
!##  bottom of the screen.  It has the same border.
!#########################################################
  Check_Erc (InitVidFrame (1, 0, 2, 80, nLines - 3,
                           0, ' ', 0, FALSE, FALSE));
!#########################################################
!##  Frame 2 overlaps frame 1 completely, but also
!##  includes the last line on the screen.
!#########################################################
  Check_Erc (InitVidFrame (2, 0, 2, 80, nLines - 2,
                           0, ' ', 0, FALSE, FALSE));
  pVidSeg.s := 0;
  pVidSeg.r := 0;
  Check_Erc (InitCharMap (pVidSeg, sMap));
!##  Initiate video refresh  ##
  Check_Erc (SetScreenVidAttr (1, TRUE));
  Check_Erc (PosFrameCursor (2, 0, nLines - 3));
  current_col := 0;
  BannerStart := (80 - sBanner ) DIV 2;
  Check_Erc (PutFrameChars (0, BannerStart, 0,
                            ADS banner, sBanner));
END; (* PROCEDURE Screen_setup *)

!#########################################################
  PROCEDURE Process_Dcom_Input  [PUBLIC];
!#########################################################
VAR
  i     : INTEGER;
BEGIN
  Check_Erc (ScrollFrame (1, 0, 255, 1, TRUE));
  Check_Erc (PosFrameCursor (1, 255, 255));
  Check_Erc (PosFrameCursor (2, 0, nLines - 3));
  vid_col := 0;
  IF sBuf > 0
    THEN
      FOR i := 0 TO sBuf - 1 DO
        BEGIN
          Check_Erc (PutFrameChars (1, vid_col, nLines - 4,
                                    ADS buff [i], 1));
          vid_col := vid_col + 1;
          IF (vid_col > 79).
            THEN
              BEGIN
                Check_Erc (ScrollFrame (1, 0, 255, 1, TRUE));

                Check_Erc (PosFrameCursor (1, 255, 255));
                vid_col := 0;
              END;
        END;
END; (* PROCEDURE Process_Dcom_Input *)
```

```
!#########################################################
  PROCEDURE Process_Kbd_Input  [PUBLIC];
!#########################################################
BEGIN
  IF (Key = 8)    !8=BACKSPACE
    THEN
      BEGIN
        IF (current_col = 79)
          THEN Check_Erc (PutFrameChars (2, 79,
                          nLines - 3, ADS ' ', 1));
        IF (current_col > 0)
          THEN current_col := current_col - 1;
        Check_Erc (PutFrameChars (2, current_col, nLines - 3,
                                  ADS ' ', 1));
        Check_Erc (PosFrameCursor (2, current_col, nLines - 3));
      END
    ELSE
      BEGIN
        Msg [current_col] := Key;
        Check_Erc (PutFrameChars (2, current_col, nLines - 3,
                                  ADS Key, 1));
        IF (current_col < 79)
          THEN current_col := current_col + 1;
        Check_Erc (PosFrameCursor (2, current_col, nLines - 3));
      END;
END; (* PROCEDURE Process_Kbd_Input *)

!#########################################################
  PROCEDURE Active_state  [PUBLIC];
!#########################################################
!#   This is the main loop of the program.  It
!#   alternately checks the BMULTI report queue
!#   and the keyboard queue for activity.
!#########################################################
VAR
  loop1       : BOOLEAN;
  loop2       : BOOLEAN;

BEGIN
  WHILE TRUE DO
  BEGIN

    IF ASBlk.XmtStatus = 16
      THEN.
        BEGIN
          ASBlk.XmtStatus := 0;
          erc := ReadBMULTI (th, ADS sBuf, sBufMax);
        END;


    IF ASBlk.XmtStatus = 2
      THEN         BEGIN
          erc := ResetBMULTI (th);
          erc := WriteBMULTI (th, ADS Msg, cMsg);
        END;
```

```
      IF ASBlk.RcvStatus = 16
        THEN
          BEGIN
            ASBlk.RcvStatus := 0;
            Process_Dcom_Input;
            erc := ReadBMULTI (th, ADS sBuf, sBufMax);
          END;

      Erc := ReadKbdDirect (1, ADS Key);
      IF (Erc <> 802)
        THEN
          CASE Key OF

            4:  RETURN;  !FINISH key

            10, 27:       !RETURN, NEXT, and GO keys
              BEGIN
                cMsg := current_col;
                ERC := ResetBMULTI (th);
                Erc := WriteBMULTI (th, ADS Msg, cMsg);
                IF (Erc = ErcOk)
                  THEN
                    BEGIN
                      Check_Erc (ScrollFrame (2, 0, 255, 1, TRUE));
                      Check_Erc (PosFrameCursor (2, 0, nLines - 3))
                      current_col := 0;
                    END
              END

            OTHERWISE    Process_Kbd_Input;

          END;  (* CASE Key OF *)
    END;    (* WHILE TRUE DO *)

END;    (* PROCEDURE Active_state *)
!##########################################################
!#           MAIN PROGRAM
!##########################################################
BEGIN

!##########################################################
!#  Retrieve Device Address as either parameter 1 or 2,
!#  depending on whether Run File command is used, or
!#  the program's own command.
!##########################################################

  Check_Erc (RgParam (1, 0, ADS SdRet));
  IF (SdRet.sSubParam <> 2)
    THEN  Check_Erc (RgParam (2, 0, ADS SdRet));
  DevAdr := SdRet.pSubParam^;

!##########################################################
!#  Initialize the video.
!##########################################################
  Screen_setup;
```

```
!#########################################################
!#  Log onto BMULTI with the Device Address.
!#########################################################
  Erc := OpenBMULTI (DevAdr, FALSE, 80, ADS th, ADS ASBlk);
  IF (Erc <> ErcOk)
    THEN ErrorExit (Erc);
  Erc := ReadBMULTI (th, ADS sBuf, sBufMax);

!#########################################################
!#  Enter an infinite loop.
!#########################################################
  Active_state;     !Does not return until FINISH is hit.

!#########################################################
!#  At termination, deallocate resources.
!#########################################################
  Erc := CloseBMULTI (th);
  Exit;
END.        (* PROGRAM HLITerm *)
```

Pascal Terminal Program (Using Low–Level Interface)

This program uses the Low–Level Interface (LLI) to implement a simple terminal. Keyboard and screen code is shown, as well as data communications handling.

```
{$DEBUG-}
{$ENTRY-}

PROGRAM MiniTermBm;

TYPE
    String2  = STRING(2);
    pbType   = ADS OF BYTE;
    pwType   = ADS OF WORD;
    ppType   = ADS OF pbType;
    psType   = ADS OF String2;

CONST

 (* Command codes for new low-level interface *)
    Xfer_Rec_Bufc = 16#0001;
    Xfer_Xmt_Bufc = 16#0002;
    Offlinec      = 16#0003;
    Onlinec       = 16#0004;
    Idlec         = 16#0005;
    Fastsetc      = 16#0006;
    Receivec      = 16#0007;
    Transmitc     = 16#0008;
    Endsessionc   = 16#0009;
    Abortc        = 16#000A;
    Fastresetc    = 16#000B;
    Xmt_Big_Bufc  = 16#000C;

 (* Report codes *)
    No_report     = 16#0000;
    Receiving     = 16#0001;
    Rec_Grp_Sel   = 16#0002;
    Receive_Done  = 16#0003;
    Rdy_Xmt_Xfer  = 16#0004;
    Transmit_Done = 16#0005;
    Select_Denied = 16#0006;
    Receive_err   = 16#0007;
    Dup_seq_num   = 16#0008;
    Seq_num_er    = 16#0009;
    Transmit_err  = 16#000A;
    Internal_err  = 16#00FF;

(* Erc return codes for new low-level interface *)
    ErcOk               = 16#0000;
    ErcInvalidCmd       = 16#8000;
    ErcTaskOverflow     = 16#8001;
    ErcCmdPending       = 16#8002;
    ErcReportPending    = 16#8003;
    ErcInvalidAddress   = 16#8004;
```

```
    ErcCmdDenied          = 16#8005;
    ErcBufferOverflow     = 16#8006;
    ErcInvalidReportRq    = 16#8007;
    ErcReadInProgress     = 16#8008;
    ErcWriteInProgress    = 16#8009;
    ErcBufferInUse        = 16#800A;
    ErcInvalidBufLength   = 16#800B;
    ErcOfflineDenied      = 16#800C;
    ErcOnlineDenied       = 16#800D;
    ErcIdleDenied         = 16#800E;
    ErcFastRdyDenied      = 16#800F;
    ErcXmtRdyDenied       = 16#8010;
    ErcRcvRdyDenied       = 16#8011;
    ErcXfrXmtDenied       = 16#8012;
    ErcXfrRcvDenied       = 16#8013;
    ErcEndSessDenied      = 16#8014;
    ErcNotOnline          = 16#8015;
    ErcStationOverflow    = 16#8016;
    ErcAddrIsGrpAddr      = 16#8017;
    ErcIncompleteMsg      = 16#8018;
    ErcInternalError      = 16#8019;
    ErcDupVirtualAdr      = 16#801A;
    ErcReconfiguration    = 16#801B;
    ErcEntryError         = 16#801C;
    ErcStationActive      = 16#801D;

  (* Miscellaneous *)
    banner          = 'B20 Mini-Term (LLI)';
    sBanner         = 19;

VAR [PUBLIC]
    Erc             : WORD;
    Report          : WORD;
    Sh              : byte;
    DevAdr          : String2;
    DevAdrRet       : String2;
    cMsg            : INTEGER;
    Msg             : ARRAY[0..79] OF BYTE;
    sBufMax         : INTEGER;
    sBuf            : INTEGER;
    Buff            : ARRAY[0..2047] OF CHAR;
    dummyPtr        : pbType;
    pVidSeg         : pbType;
    sMap            : WORD;
    nLines          : INTEGER;
    Key             : BYTE;
    current_col     : INTEGER;
    vid_col         : INTEGER;
    SdRet           : RECORD
        pSubParam   : psType;
        sSubParam   : WORD;
                      END;
    vHdw            : RECORD.
        level       : BYTE;
        nLinesMax   : SINT;
        nColsNar    : BYTE;
        nColsWide   : BYTE;
                      END;
```

```
VALUE
    sBufMax         := 4096;
    sMap            := 16#0B0C;

!#########################################################
!#          System Common Procedures
!#########################################################
PROCEDURE Exit; EXTERN;

PROCEDURE ErrorExit (errTerm : WORD); EXTERN;

FUNCTION  PosFrameCursor (iFrame : INTEGER;
                          iCol   : INTEGER;
                          iLine  : INTEGER) : WORD; EXTERN;

FUNCTION  PutFrameChars (iFrame : INTEGER;
                         iCol   : INTEGER;
                         iLine  : INTEGER;
                         pbText : pbType;
                         cbText : INTEGER) : WORD; EXTERN;

FUNCTION  ResetFrame ( iFrame : INTEGER) : WORD; EXTERN;

FUNCTION  ScrollFrame ( iFrame     : INTEGER;
                        iLineStart : INTEGER;
                        iLineMax   : INTEGER;
                        cLines     : INTEGER;
                        fUp        : BOOLEAN) : WORD; EXTERN;

!#########################################################
!#          Object Module Procedures
!#########################################################
FUNCTION  BmOpenII (Addr     : string2;
                    pSh      : pbType;
                    fSys     : boolean;
                    pAddrRet : pwType;
                    pbNode   : pbType;
                    cbNode   : word;
                    pbDesc   : pbType;
                    cbDesc   : word) : WORD; EXTERN;

FUNCTION BmReportTimeout (TaskH      : WORD;
                          pReportRet : pwType;
                          timeout    : WORD) : WORD; EXTERN;

FUNCTION BmReportWait (TaskH      : WORD;
                       pReportRet : pwType) : WORD; EXTERN;

FUNCTION BmReport (TaskH      : WORD;
                   pReportRet : pwType) : WORD; EXTERN;

FUNCTION BmCommand (TaskH   : WORD;
                    Comm    : WORD;
                    pbuffer : pbType;
                    sbuffer : INTEGER) : WORD; EXTERN;

FUNCTION  RgParam (iParam    : WORD;
                   iSubParam : WORD;
                   pSdRet    : pbType) : WORD; EXTERN;
```

```
!#############################################################
!#          Procedural requests
!#############################################################
FUNCTION  InitCharMap (pMap : pbType;
                       sMap : WORD) : WORD; EXTERN;

FUNCTION  InitVidFrame (iFrame      : INTEGER;
                        iColStart   : INTEGER;
                        iLineStart  : INTEGER;
                        nCols       : INTEGER;
                        nLines      : INTEGER;
                        borderDesc  : BYTE;
                        bBorderChar : CHAR;
                        bBorderAttr : BYTE;
                        fDblHigh    : BOOLEAN;
                        fDblWide    : BOOLEAN) : WORD; EXTERN;

FUNCTION  QueryVidHdw (pBuf : pbType;
                       sBuf : WORD) : WORD; EXTERN;

FUNCTION  ReadKbdDirect (mode     : WORD;
                         pCharRet : pbType) : WORD; EXTERN;

FUNCTION  ResetVideo (nCols    : INTEGER;
                      nLines   : INTEGER;
                      fAttr    : BOOLEAN;
                      bSpace   : CHAR;
                      psMapRet : pbType) : WORD; EXTERN;

FUNCTION  SetScreenVidAttr (iAttr : WORD;
                            fOn   : BOOLEAN) : WORD; EXTERN;

!#############################################################
  PROCEDURE Check_Erc (irk : WORD)  [PUBLIC];
!#############################################################
BEGIN
  IF (irk <> 0) THEN ErrorExit (irk);
END;

!#############################################################
  PROCEDURE Screen_setup  [PUBLIC];
!#############################################################
!#   Switches the video map back and forth.
!#############################################################
VAR
  BannerStart : INTEGER;
BEGIN
  Check_Erc (QueryVidHdw (ADS vHdw, 4));
  nLines := vHdw.nLinesMax;
  Check_Erc (ResetVideo (80, nLines, FALSE,
                         ' ', ADS sMap));
  Check_Erc (InitVidFrame (0, 0, 0, 80, 1,
                           4, '_', 0, FALSE, FALSE));
  Check_Erc (InitVidFrame (1, 0, 2, 80, nLines - 3,
                           0, ' ', 0, FALSE, FALSE));
  Check_Erc (InitVidFrame (2, 0, 2, 80, nLines - 2,
                           0, ' ', 0, FALSE, FALSE));
  pVidSeg.s := 0;
  pVidSeg.r := 0;
```

```
  Check_Erc (InitCharMap (pVidSeg, sMap));
  Check_Erc (SetScreenVidAttr (1, TRUE));
  Check_Erc (PosFrameCursor (2, 0, nLines - 3));
  current_col := 0;
  BannerStart := (80 - sBanner ) DIV 2;
  Check_Erc (PutFrameChars (0, BannerStart, 0,
                            ADS banner, sBanner));
END; (* PROCEDURE Screen_setup *)

!#######################################################
  PROCEDURE Process_Dcom_Input  [PUBLIC];
!#######################################################
VAR
  I    : INTEGER;
BEGIN
  Check_Erc (ScrollFrame (1, 0, 255, 1, TRUE));
  Check_Erc (PosFrameCursor (1, 255, 255));
  vid_col := 0;
  FOR I := 0 TO sBuf - 1 DO
    BEGIN
      Check_Erc (PutFrameChars (1, vid_col, nLines - 4,
                                ADS buff [I], 1));
      vid_col := vid_col + 1;
      IF (vid_col > 79)
        THEN
          BEGIN
            Check_Erc (ScrollFrame (1, 0, 255, 1, TRUE));
            Check_Erc (PosFrameCursor (1, 255, 255));
            vid_col := 0;
          END;
    END;
END; (* PROCEDURE Process_Dcom_Input *)

!#######################################################
  PROCEDURE Process_Kbd_Input  [PUBLIC];
!#######################################################
BEGIN.
  IF (Key = 8)    !8=BACKSPACE
    THEN
      BEGIN
        IF (current_col = 79)
          THEN Check_Erc (PutFrameChars (2, 79,
                          nLines - 3, ADS ' ', 1));
        IF (current_col > 0)
          THEN current_col := current_col - 1;
        Check_Erc (PutFrameChars (2, current_col, nLines - 3,
                                  ADS ' ', 1));
        Check_Erc (PosFrameCursor (2, current_col, nLines - 3));
      END
    ELSE
      BEGIN
        Msg [current_col] := Key;
        Check_Erc (PutFrameChars (2, current_col, nLines - 3,
                                  ADS Key, 1));
        IF (current_col < 79)
          THEN current_col := current_col + 1;
        Check_Erc (PosFrameCursor (2, current_col, nLines - 3));
      END;
END; (* PROCEDURE Process_Kbd_Input *)
```

```
!************************************************************
  PROCEDURE Active_state  [PUBLIC];
!************************************************************
!*   This is the main loop of the program.  It
!*   alternately checks the Bmulti report queue
!*   and the keyboard queue for activity.
!************************************************************
VAR
  loop1       : BOOLEAN;
  loop2       : BOOLEAN;

BEGIN
  WHILE TRUE DO
  BEGIN
    loop1 := TRUE;
    WHILE loop1 DO
    BEGIN
      Erc := BmReportTimeout (Sh, ADS Report, 2);
       IF (Erc = ErcOk)
         THEN
          CASE Report OF

            No_report:  loop1 := FALSE;

            Transmit_Done:
                Erc := BmCommand (Sh, Receivec, dummyPtr, 0);

            Rdy_Xmt_Xfer:
              Erc := BmCommand (Sh, Xfer_Xmt_Bufc,
                                ADS Msg, cMsg);

            Receive_Done, Dup_seq_num, Seq_num_err:
              BEGIN
                Erc := BmCommand (Sh, Xfer_Rec_Bufc,
                                  ADS sBuf, sBufMax);
                IF (Erc = ErcOk)
                  THEN
                    Process_Dcom_Input;
                Erc := BmCommand (Sh, Receivec, dummyPtr, 0);
              END;
          END;     (* CASE Report OF *)
    END;    (* WHILE loop1 DO *)

    loop2 := TRUE;
    WHILE loop2 DO
    BEGIN
      Erc := ReadKbdDirect (1, ADS Key);
      IF (Erc = 602)
        THEN loop2 := FALSE
        ELSE
          BEGIN
            Erc := BmCommand (Sh, Idlec, dummyPtr, 0);
            CASE Key OF

            4:  RETURN;  !FINISH key

            10, 27:      !RETURN, NEXT, and GO keys
```

```
                BEGIN
                  cMsg := current_col;
                  Erc := BmCommand (Sh, Idlec, dummyPtr, 0);
                  Erc := BmCommand (Sh, Transmitc, dummyPtr, 0);
                  IF (Erc = ErcOk)
                    THEN
                      BEGIN
                        Check_Erc (ScrollFrame (2, 0, 255, 1, TRUE));
                        Check_Erc (PosFrameCursor (2, 0, nLines - 3));
                        current_col := 0;
                      Erc := BmCommand (Sh, Receivec, dummyPtr, 0);

                 END
                END

            OTHERWISE   Process_Kbd_input;
           END;

         END;  (* CASE Key OF *)
    END;   (* loop2  *)
  END;   (* WHILE TRUE DO *)

END;   (* PROCEDURE Active_state *)
!#############################################################
!#          MAIN PROGRAM
!#############################################################
BEGIN

!#############################################################
!#  Retrieve Device Address as either parameter 1 or 2,
!#  depending on whether Run File command is used, or
!#  the program's own command.
!#############################################################
  Check_Erc (RgParam (1, 0, ADS SdRet));
  IF (SdRet.sSubParam <> 2)
    THEN  Check_Erc (RgParam (2, 0, ADS SdRet));
  DevAdr := SdRet.pSubParam?;

!#############################################################
!#  Initialize the video.
!#############################################################
  Screen_setup;

!#############################################################
!#  Log onto Bmulti with the Device Address.
!#############################################################
  Erc := BmOpenII (DevAdr, ADS Sh, FALSE, ads DevAdrRet,
                   ads nil, 0, ads 'LLITerm', 7);
  IF (Erc <> ErcOk)
    THEN ErrorExit (Erc);
  Erc := BmCommand (Sh, Onlinec, dummyPtr, 0);
```

```
!##########################################################
!#  Enter an infinite loop.
!##########################################################
  Active_state;     !Does not return until FINISH is hit.

!##########################################################
!#  At termination, deallocate resources.
!##########################################################
  REPEAT
    Erc := BmCommand (Sh, Idlec, dummyPtr, 0);
  UNTIL (Erc = ErcOk);
  Erc := BmCommand (Sh, Endsessionc, dummyPtr, 0);
  Exit;
END.        (* PROGRAM MiniTermBm *)
```

Glossary

A

ACK (Acknowledgment, 06h)
An affirmative response to a normal selection (indicating Ready to Receive) or a transmission (indicating message accepted).

AD1, AD2 (Address 1, Address 2)
A two-character address established as the address of a terminal. These characters are used to address a terminal in polling or selection or in the message heading, and identify the terminal from which a message is transmitted. Upon receipt of a message, the receiving station may use AD1-AD2 to verify that the message originated at the polled terminal. AD1 and AD2 are represented by any characters from columns 2, 3, 4, 5, 6, or 7, of the ASCII code chart in Appendix B, Figure B-2, except the character DEL, column 7, row 15, shown as 7/15.

Application Status Block (ASB)
Used by the High-Level Interface (HLI) process to communicate the status of the reads and writes issued by an application. (See Section 10 for more information and the format of the ASB.)

ASB
See Application Status Block.

ASCII
The American Standard Code for Information Interchange, ASCII defines the character set codes for information transfer between equipment.

B

BCC (Block Check Character)
A redundant character added to the end of a message for error detection and control. BCC is formed by taking a binary sum without carry, on each of the seven bits of the transmitted characters following SOH, including ETX, but excluding any SYN characters. The correct value of the character parity bit of the BCC is that which makes the sense of character parity the same as for text characters. BCC immediately follows ETX.

BMULTI
A system service used by CTOS workstations and shared resource processors to communicate with Unisys A–Series and V–Series mainframe systems.

C

CON (Contention, 07h)
A character used to instruct all terminals that receive the instruction to go to the contention mode. NULL characters replace AD1-AD2 in the contention sequence. There is no acknowledgment of the contention instruction.

configuration
A process of entering specific information (in the form of parameter values) about your hardware, software, and network arrangement to BMULTI or another program so that the program operates correctly on your system.

configuration file
The BMULTI configuration file contains the parameters used by BMULTI during its operation as a system service. The configuration file can be modified by using the BMULTI Configuration File Editor.

control station

A station on a data link that is responsible for polling, selecting, and ensuring the orderly operation of that link. (Usually a control station is a large host computer.) The control station is responsible for initiating recovery procedures during abnormal conditions on the link. All stations on a multipoint network, other than the control station, are called terminal stations. These are usually terminals but can be microcomputers (such as CTOS workstations) or minicomputers.

D

deinstalling BMULTI

The process by which BMULTI is removed from a CTOS system as a system service.

DTS

Data Transfer Service.

E

ENQ (Inquiry, 05h)

A reply request control character. ENQ is used as the final character of a poll or of a select, when a response is required from the other station.

EOT (End of Transmission)

A character transmitted by a terminal as a "no traffic" response to a poll. Receipt of EOT places the terminal in a control state listening for a polling or selection sequence. EOT may be transmitted instead of ETX to abort a transmission

F

FSL (Fast Select, 73h)

A character used to indicate a Fast select in a selection sequence transmitted by the central computer. Fast select is followed immediately by a message without requiring acknowledgment of the selection.

G

GSL (Group Select)

A character used to indicate a message for a group of stations. In the group select sequence, AD1–AD2, identifies the station that is to acknowledge receipt of the message. Group select is followed immediately by a message without requiring acknowledgment of the selection. Group select may be represented by any agreed–upon character selected from columns two through six of the ASCII chart (Appendix B).

H

High-Level Interface (HLI)

A procedural interface to BMULTI intended primarily for application programmers who do not need to concentrate on the details of the BMULTI interface.

I

installing BMULTI

The process by which BMULTI is loaded on a CTOS workstation, IDS module, or shared resource processor as a system service.

L

Low-Level Interface (LLI)

A procedural interface to BMULTI with a high degree of data communications control. It is intended to be used by applications needing access to advanced BMULTI communications features and services, including the BMULTI Suspend/Resume filter.

N

NAK (Negative Acknowledgment, 15h)

A negative response to a selection (indicating Not Ready to Receive) or a transmission (indicating character parity failure for any character in a message or a failure of the BCC).

P

parameters
Values used by a program to define its operation. In the case of BMULTI, configuration file parameters are used to tell BMULTI how to communicate with the host.

POL (Poll, 70h)
Indicates a poll preceding ENQ in a polling sequence.

R

Release Notes file
Contains information about BMULTI features and last minute update information. The Release Notes file is distributed on the BMULTI software installation disk.

ring buffer
A data buffer in which the newest information overwrites the oldest. The information does not "overflow," but rather replaces the information at the beginning of the buffer.

RVI (Reverse Interrupt (DLEG) 103Ch)
Sent by the control station instead of a positive acknowledgment (ACK) when the control station has priority messages to deliver. RVI is normally used in a group poll environment to requrest premature termination of a series of message transmissions or to poll other terminals. On receipt of an RVI, the terminal should send EOT as soon as possible.

S

SEL (Select, 71h)
Indicates a normal select preceding ENQ in a selection sequence.

Service ID
Used with the Multiple Gateways Service (MGS), the Service ID identifies a gateway, or in the case of BMULTI, a specific copy of the BMULTI system service running on a particular processor. Service IDs are defined for BMULTI in the BMULTI configuration files. Applicatior access the gateway by specifying the Service ID in the routing information of all requests that direct BMULTI to establish a connectior (for example, the BMULTI LLI call, BmOpenII).

Shift-In/Shift-Out (SI/SO)
A character sequence that allows BMULTI, which uses a 7–bit characte to use the entire 8–bit ASCII character set.

Software Installation
The method through which files are copied from the product distributior disks onto the hard disk of a CTOS workstation or shared resource processor, request codes are updated, and new commands are created to use the new software.

SOH (Start of Heading, 01h)
The first of a sequence of characters that form the heading. The headin also contains a terminal identification (AD1, AD2) and may contain transmission numbers (XMno). A heading is ended by STX.

status monitor
A utility that displays the current status of BMULTI.

STX (Start of Text, 02h)
Terminates a heading, and precedes a sequence of characters that form the text of the transmission.

SYN (Synchronous Idle, 16h)

Used only with synchronous transmission in the absence of any other character to provide a signal for establishing and retaining synchronism. Upon initiating a synchronous transmission, several SYN characters are transmitted before the transmission of any other character. This enables the receiving station to acquire character synchronization. SYN is also used as a time fill when no other characters are available for transmission at any point in a character sequence, except between ETX and the next following BCC. SYN is purged at the receiving station and is not included in the summation for BCC.

system service

Software that is loaded into the RAM memory of a workstation, IDS module, or shared resource processor. Once loaded, it handles requests for its whatever services it can provide. The BMULTI Agent, when configured and loaded into RAM, is a system service.

T

transmission number

A number identifying, in sequence, transmissions to or from a terminal. It is used optionally as part of a message header to assist in message error detection and recovery. Separate sets of transmission numbers are used for broadcast and group addressed messages.

V

Virtual Address (VAD)

The method by which BMULTI avoids timing out inactive stations on the host computer. VADs act like active addresses in the Idle state.

Index

A

B

C

D

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

X