MPX-32 ™

Installation and System Administration

Revision 3.5

Reference Manual Volume III

April 1990

ENCORE

323-001553-600

Limited Rights

This manual is supplied without representation or warranty of any kind. Encore Computer Corporation therefore assumes no responsibility and shall have no liability of any kind arising from the supply or use of this publication or any material contained herein.

Proprietary Information

The information contained herein is proprietary to Encore Computer Corporation and/or its vendors, and its use, disclosure, or duplication is subject to the restrictions stated in the standard Encore Computer Corporation License terms and conditions or the appropriate third-party sublicense agreement.

Restricted Rights

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at 252.227.7013.

Encore Computer Corporation
6901 West Sunrise Boulevard
Fort Lauderdale, Florida 33313

History

The MPX-32 Release 3.0 Reference Manual, Publication Order Number **323-001550-000**, was printed June, 1982.

Publication Order Number **323-001553-100**, (Revision 1, Release 3.2) was printed June, 1983.

Publication Order Number **323-001553-200**, (Revision 2, Release 3.2B) was printed March, 1985.

Publication Order Number **323-001553-201**, (Change 1 to Revision 2, Release 3.2C) was printed December, 1985.

Publication Order Number **323-001553-300**, (Revision 3, Release 3.3) was printed December, 1986.

Publication Order Number **323-001553-400**, (Revision 4, Release 3.4) was printed January, 1988.

Publication Order Number **323-001553-500**, (Revision 3.4U03) was printed October, 1989.

Publication Order Number **323-001553-600**, (Revision 3.5) was printed April, 1990.

This manual contains the following pages:

Title page
Copyright page
iii/iv through xxiii/xxiv
1-1 through 1-2
2-1 through 2-27/2-28
3-1 through 3-5/3-6
4-1 through 4-9/4-10
5-1 through 5-5/5-6
6-1 through 6-4
7-1 through 7-66
8-1 through 8-33/8-34
9-1 through 9-11/9-12
10-1 through 10-62
11-1 through 11-15/11-16
12-1 through 12-17/12-18
13-1 through 13-23/13-24
14-1 through 14-6
A-1 through A-7/A-8
B-1 through B-42
C-1 through C-32
D-1 through D-2
E-1 through E-2
F-1 through F-2
G-1 through G-2
H-1 through H-2
I-1/I-2
J-1 through J-2
K-1/K-2
L-1 through L-79/L-80
GL-1 through GL-10
IN-1 through IN-31/IN-32

Contents

Page

4 Installing a User-Configured System

5 Online Restart

6 Recovering the System

7 System Generation (SYSGEN)

Page

Page

Page

Contents

Page

8 System Debugger

Page

Page

Page

11 Device Initializer/Loader (DEVINITL)

12 Alterable Control Store (ACS)

Contents

List of Figures

List of Tables

Documentation Conventions

Conventions used in directive syntax, messages, and examples throughout the MPX-32 documentation set are described below.

Messages and Examples

Text shown in `this distinctive font` indicates an actual representation of a system message or an example of actual input and output. For example,

```
VOLUME MOUNT SUCCESSFUL
```

or

```
TSM>!ACTIVATE MYTASK
TSM>
```

Lowercase Italic Letters

In directive syntax, lowercase italic letters identify a generic element that must be replaced with a value. For example,

$NOTE *message*

means replace *message* with the desired message. For example,

```
$NOTE 10/12/89 REV 3
```

In system messages, lowercase italic letters identify a variable element. For example,

`**BREAK** ON:`*taskname*

means a break occurred on the specified task.

Uppercase Letters

In directive syntax, uppercase letters specify the input required to execute that directive. Uppercase bold letters indicate the minimum that must be entered. For example,

$ASSIGN *lfc* **TO** *resource*

means enter $AS or $ASSIGN followed by a logical file code, followed by TO and a resource specification. For example,

```
$AS OUT TO OUTFILE
```

In messages, uppercase letters specify status or information. For example,

```
TERMDEF HAS NOT BEEN INSTALLED
```

Documentation Conventions

Brackets [ ]

An element inside brackets is optional. For example,

$CALL *pathname* [*arg*]

means supplying an argument (*arg*) is optional.

Multiple items listed within brackets means enter one of the options or none at all. The choices are separated by a vertical line. For example,

$SHOW [**CPU**TIME | **JOB**S | **USE**RS]

means specify one of the listed parameters, or none of them to invoke the default.

Items in brackets within encompassing brackets or braces can be specified only when the other item is specified. For example,

BACKSPACE **F**ILE [[**FIL**ES=] *eofs*]

indicates if *eofs* is supplied as a parameter, FIL= or FILES= can precede the value specified.

Commas within brackets are required only if the bracketed element is specified. For example,

LIST [*taskname*][,[*ownername*] [,*pseudonym*]]

indicates that the first comma is required only if *ownername* and/or *pseudonym* is specified. The second comma is required only if *pseudonym* is specified.

Braces { }

Elements listed inside braces specify a required choice. Choices are separated by a vertical line. Enter one of the arguments from the specified group. For example,

[**BLO**CKED={**Y** | **N**}]

means Y or N must be supplied when specifying the BLOCKED option.

Horizontal Ellipsis ...

The horizontal ellipsis indicates the previous element can be repeated. For example,

$DEFM [*par*] [,*par*] ...

means one or more parameters (*par*) separated by commas can be entered.

Vertical Ellipsis

The vertical ellipsis indicates directives, parameters, or instructions have been omitted. For example,

```
$DEFM SI,ASSEMBLE,NEW,OP
     .
     .
     .
$IFA %OP ASSM
```

means one or more directives have been omitted between the $DEFM and $IFA directives.

Parentheses ( )

In directive syntax, parentheses must be entered as shown. For example,

(*value*)

means enter the proper value enclosed in parentheses; for example, (234).

Special Key Designations

The following are used throughout the documentation to designate special keys:

<ctrl>	control key
<ret> or <CR>	carriage return/enter key
<tab>	tab key
<break>	break key
<bck>	backspace key
<del>	delete key

When the <ctrl> key designation is used with another key, press and hold the control key, then press the other key. For example,

<ctrl>C

means press and hold the control key, then press the C.

Change Bars

Change bars are vertical lines (|) appearing in the right-hand margin of the page for your convenience in identifying the changes made in MPX-32 Revision 3.5.

When an entire chapter has been changed or added, change bars appear at the chapter title only. When text within figures has changed, change bars appear only at the top and bottom of the figure box.

1 Building and Maintaining the System

1.1 Introduction

MPX-32 uses system utilities such as the Volume Manager and Text Editor to provide mechanisms for building and maintaining resident operating systems. A resident system is configured by running the System Generator utility, SYSGEN. A tailored system is configured by modifying the Master System Distribution Tape (SDT) and SYSGEN file before installation. A System Debugger is supplied that can debug a resident operating system or resident user-developed interrupt and device handlers.

This volume provides documentation on installation from a System Distribution Tape (SDT) and Utility Tape, SYSGEN, online and IOP console restart capability, the System Debugger, the System Patch facility, System Administrator Services, the Device Initializer/Loader, the Alterable Control Store Load and Display utility, the Volume Formatter, and Volume Compress.

The MPX-32 operating system supports floppy disk usage. All references to the System Distribution Tape (SDT) apply whether the distribution medium is magnetic tape or floppy disk.

Figure 1-1 provides an overview of installation and configuration as described in Chapters 2 through 4 of this volume.

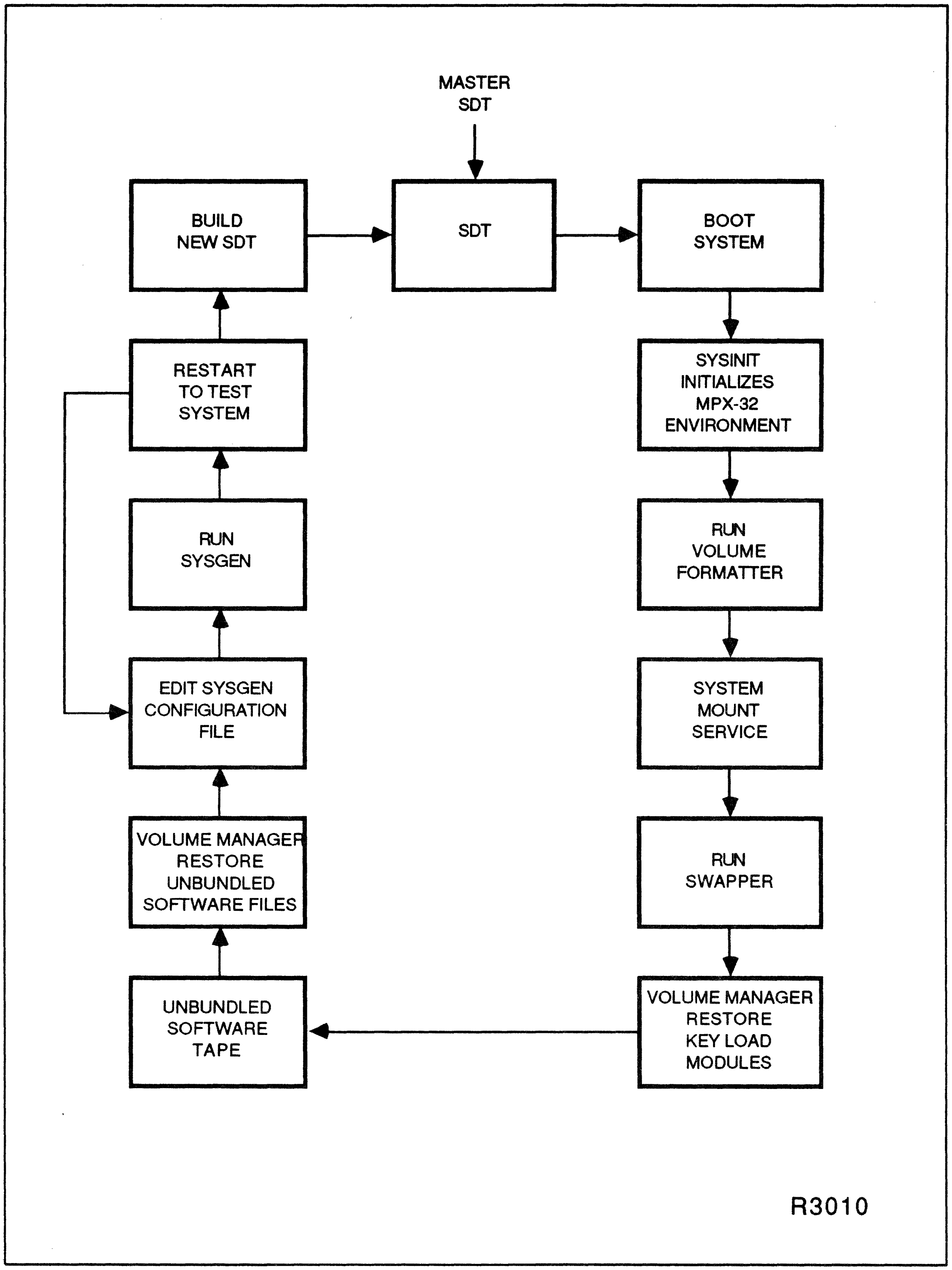

Figure 1-1
MPX-32 Installation/Configuration Overview

2 Installing a Starter System

2.1 Introduction

Starter systems are supplied on the Master System Distribution Tape (SDT). This chapter describes the minimum hardware configuration supported by the starter system and the format of the Master SDT. It also includes an example of booting the starter system.

2.2 Hardware Configuration

The following hardware and logical addresses are used for installing the starter system on a CONCEPT/32 computer:

	Software	Hardware	
		Channel	Subaddress
128KW Memory			
XIO magnetic tape	*xxxx*	*xx*	*xx*
MFP SCSI tape	7E40	7E	40
MFP/IOP console -or-	7EFC	7E	FC
CONCEPT 32/2000 on-board CPU console	02FC	02	FC
MFP/IOP line printer	7EF8	7E	F8
Disk drive (XIO) (Class F)	*xxxx*	*xx*	*xx*

User-definable addresses are indicated by *x*.

Only the system console is configured in the starter system. User terminals are configured by the SYSGEN utility. Once terminals have been configured, they can be initialized by the system module J.TINIT. If a terminal initialization file called LOGONFLE has not yet been created, all terminals are marked online, but noted as failed devices (dead terminals). Before LOGONFLE is created, all terminals have the following defaults:

Wakeup Character - ?
Baud Rate - 9600
Parity - EVEN
Character Size - 7
ALIM Only - HALF DUPLEX

For details on creating a LOGONFLE file, see Chapter 10 of this volume.

The following disks are supported by MPX-32 and can be used when booting from the Master SDT:

Disk	SYSGEN Type Code	J.VFMT Disk Parameter Type Code
80MB moving head disk - UDP/DPII	MH080	DC0080
160MB moving head disk - UDP/DPII	MH160	DC0160
300MB moving head disk - UDP/DPII	MH300	DC0300
340MB moving head disk - UDP/DPII	MH340	DC0340
600MB moving head disk - UDP/DPII	MH600	DC0600
80MB cartridge module disk - HSDP	ANY	DP0080
337MB moving head disk - HSDP	ANY	DP0337
474MB moving head disk - HSDP	ANY	DP0474
500MB moving head disk - HSDP	ANY	DP0500
689MB moving head disk - HSDP	ANY	DP0689
800MB moving head disk - HSDP	ANY	DP0800
850MB moving head disk - HSDP	ANY	DP0850
858MB moving head disk - HSDP	ANY	DP0858
1230MB moving head disk - HSDP	ANY	DP01230
150MB moving head disk - SCSI	ANY	SD0150
300MB moving head disk - SCSI	ANY	SD0300
700MB moving head disk - SCSI	ANY	SD0700
Any other non-floppy disk	ANY	NOT AVAILABLE

* Disk code applies to both IOP disks and disk processors.

If a disk code is specified and a mismatch with the drive occurs, an error results. If disk code ANY is specified with a nonfloppy disk, a mismatch error cannot occur since the relevant operating system table entries are modified to reflect the drive.

2.3 The Master System Distribution Tape (SDT)

The system initialization (SYSINIT) process (see section 2.6, System Builder) distinguishes between a Master and user SDT by the system name stored in the communications region of the MPX-32 operating system (C.SYSTEM). Therefore, the following file names are reserved for the system: MSTRALL; MSTREXT; and MSTROUT.

When MSTRALL, MSTREXT, or MSTROUT is detected by SYSINIT, the processing of the system images for a master SDT boot is performed. Therefore, these files should not be modified.

Following the system images are a group of saved files. These files include all essential load modules to support a fully operational MPX-32 system. Also included are the object files required to SYSGEN an MPX-32 system tailored to individual hardware and software requirements.

Figure 2-1 shows the format of the Master SDT. Table 2-1 lists the software included on the Master SDT.

Table 2-1
Deliverable Software for MPX-32

Operating System Modules

The following modules are memory resident:

Source File Name	Object File Name	Program Name	Description
SH.ACBA	OH.ACBA	H.ACBA	Vector Processor SVC
SH.ADA	OH.ADA*	H.ADA	Aplex ADA Support
SH.ALOC	OH.ALOC*	H.ALOC	Resource Allocation (Compatible Interface)
SH.BKDM	OH.BKDM*	H.BKDM	Blocked Data Management Module
SH.DBUG1	OH.DBUG1*	H.DBUG1	Mapped Portion of System Debugger(for resident O.S.)
SH.DBUG2	OH.DBUG2*	H.DBUG2	Unmapped portion of System Debugger (for resident O.S.)†
SH.DMPMT	OH.DMPMT*	H.DMPMT	Stand-alone Dump Writer
SH.EXEC	OH.EXEC*	H.EXEC	Executive (CPU & IPU Support)
SH.EXEC2	OH.EXEC2*	H.EXEC2	Optional Executive (CPU & IPU DELTA Support)
SH.EXEC3	OH.EXEC3*	H.EXEC3	Optional Executive (CPU Only)
SH.EXSUB	OH.EXSUB*	H.EXSUB	Common Executive Subroutines
SH.FISE	OH.FISE*	H.FISE	File System (Compatible Interface)
SH.IOCS	OH.IOCS*	H.IOCS	Input/Output Control System
SH.MDT	OH.MDT*	H.MDT	Rapid File Access Module
SH.MEMM	OH.MEMM*	H.MEMM	Memory Management Module
SH.MEMM2	OH.MEMM2*	H.MEMM2	Optional Memory Management Module
SH.MONS	OH.MONS*	H.MONS	Monitor Services (Compatible Interface)
SH.MVMT	OH.MVMT*	H.MVMT	Multivolume Magnetic Tape Management
SH.PET	OH.PET**	H.PET	Performance Evaluation Module
SH.PTRAC	OH.PTRAC*	H.PTRAC	PTRACE Module
SH.REMM	OH.REMM*	H.REMM	Resource Management Module
SH.REXS	OH.REXS*	H.REXS	Resident Executive Services
SH.SINIT	OH.SINIT*	H.SINIT	System Initializer
SH.SURE	OH.SURE*	H.SURE	High Performance Suspend/Resume Module
SH.SWAPR	OH.SWAPR*	H.SWAPR	Swapper (Resident)
SH.TAMM	OH.TAMM*	H.TAMM	Task Management Module
SH.TDEF	OH.TDEF	H.DEF	Terminal Definition Module
SH.TSM	OH.TSM*	H.TSM	Terminal Service Manager
SH.VOMM	OH.VOMM*	H.VOMM	Volume Management Module

† Remains physically memory resident but is not included in logical address space as part of the system map.

* These modules reside in directory OBJECT_OUT as well as OBJECT on the Master SDT.

** This module resides in directory PET on the Master SDT.

Extended Operating System Modules

The following modules are Extended MPX-32 modules located in directory OBJECT_E.

OH.ALOC	Extended H.ALOC
OH.EXSUB	Extended H.EXSUB
OH.FISE	Extended H.FISE
OH.MEMM	Extended H.MEMM
OH.MONS	Extended H.MONS
OH.PTRAC	Extended H.PTRAC
OH.REMM	Extended H.REMM
OH.REXS	Extended H.REXS
OH.TAMM	Extended H.TAMM
OH.TSM	Extended H.TSM
OH.VOMM	Extended H.VOMM

Interrupt and Trap Handlers

The following routines are memory resident:

Source File Name	Object File Name	Program Name	Description
SH.CALM	OH.CALM*	H.CALM	Optional Calm Replacement SVC Trap Processor
SH.CPU	OH.CPU*	H.CPU	IPU to CPU Trap Processor (IPU Task Scheduler)
SH.CPU2	OH.CPU2*	H.CPU2	Optional IPU to CPU Trap Processor (IPU Task Scheduler)
SH.ICP	OH.ICP*	H.ICP	Indirectly Connected Interrupt Program
SH.IP00	OH.IP00*	H.IP00	Power Fail Trap Processor
SH.IP02	OH.IP02*	H.IP02	Memory Parity Trap Processor
SH.IP03	OH.IP03*	H.IP03	Nonpresent Memory Trap Processor
SH.IP04	OH.IP04*	H.IP04	Undefined Instruction Trap Processor
SH.IP05	OH.IP05*	H.IP05	Privilege Violation Trap Processor
SH.IP06	OH.IP06*	H.IP06	SVC Trap Processor
SH.IP07	OH.IP07*	H.IP07	Machine Check Trap Processor
SH.IP08	OH.IP08*	H.IP08	System Check Trap Processor
SH.IP09	OH.IP09*	H.IP09	MAP Fault Trap Processor
SH.IP0C	OH.IP0C*	H.IP0C	Address Specification Trap Processor (32/87, 32/97)
SH.IP0F	OH.IP0F*	H.IP0F	Arithmetic Exception Trap Processor
SH.IP10	OH.IP10*	H.IP10	Cache Memory Parity Error Trap Processor
SH.IP13	OH.IP13*	H.IP13	Attention Interrupt Processor
SH.IPAS	OH.IPAS*	H.IPAS	System Auto-start Trap Processor
SH.IPCL	OH.IPCL*	H.IPCL	Real-time Clock Interrupt Processor
SH.IPHT	OH.IPHT*	H.IPHT	CPU Halt Trap Handler
SH.IPIT	OH.IPIT*	H.IPIT	Interval Timer Interrupt Processor
SH.IPPF	OH.IPPF*	H.IPPF	Page Fault Trap Processor
SH.IPU	OH.IPU*	H.IPU	IPU Executive Trap Processor
SH.IPUAS	OH.IPUAS*	H.IPUAS	IPU Power Up Auto Start Trap Processor
SH.IPUIT	OH.IPUIT*	H.IPUIT	IPU Accounting Interval Timer Processor
SH.IPVP	OH.IPVP*	H.IPVP	Vector Processor Interrupt Handler

* These modules reside in directory OBJECT_OUT as well as OBJECT on the Master SDT.

Device Handlers

The following handlers are memory resident:

Source File Name	Object File Name	Load Module	Description
SH.ASMP	OH.ASMP	H.ASMP	ALIM (ASYNC) - GPMC
SH.BSMP	OH.BSMP	H.BSMP	BLIM (BISYNC) - GPMC
SH.CPMP	OH.CPMP	H.CPMP	Card Reader/Punch - GPMC
SH.CTXIO	OH.CTXIO*	H.CTXIO	IOP Console Terminal
SH.DCSCI	OH.DCSCI*	H.DCSCI	MFP Disk Handler
SH.DCXIO	OH.DCXIO*	H.DCXIO	XIO Disk Handler
SH.DPXIO	OH.DPXIO*	H.DPXIO	HSDP Disk Handler
SH.F8XIO	OH.F8XIO*	H.F8XIO	IOP 8-Line ASYNC (Full duplex support)
SH.GPMCS	OH.GPMCS	H.GPMCS	GPMC Subroutines
SH.HSDG	OH.HSDG*	H.HSDG	Generic High Speed Data (HSD)
SH.IBLG	OH.IBLG*	H.IBLG	Inter-Bus Link Handler
SH.IFXIO	OH.IFXIO*	H.IFXIO	XIO Channel Interrupt Fielder
SH.LPXIO	OH.LPXIO*	H.LPXIO	XIO Line Printer
SH.MDXIO	OH.MDXIO*	H.MDXIO	XIO Memory Disk
SH.MTSCI	OH.MTSCI*	H.MTSCI	MFP Tape Handler
SH.MTXIO	OH.MTXIO*	H.MTXIO	XIO Magnetic Tape
SH.MUX0	OH.MUX0	H.MUX0	GPMC Multiplexer
SH.NUXIO	OH.NUXIO*	H.NUXIO	XIO Null Device
SH.SLMP	OH.SLMP	H.SLMP	SLIM (Synchronous) - GPMC
SH.XIOS	OH.XIOS*	H.XIOS	XIO Common Subroutines

* These modules reside in directory OBJECT_OUT as well as OBJECT on the Master SDT.

Nonresident Routines

Source File Name	Object File Name	Load Module	Description
SJ.ACCNT	OJ.ACCNT	J.ACCNT	Accounting Utility
SJ.ADMNT	OJ.ADMNT	ADMOUNT	Dismount ANSI Labeled Tape Utility
SJ.AMOUNT	OJ.AMOUNT	AMOUNT	Mount ANSI Labeled Tape Utility
SJ.ASTAT	OJ.ASTAT	ASTAT	Display ANSI Labeled Tape Utility
SJ.ATAPE	OJ.ATAPE	J.ATAPE	ANSI Labeled Tape Processing Task
SJ.AUTO	OJ.AUTO	N/A	Auto Disk Geometry Subroutine
SJ.AVOL1	OJ.AVOL1	AVOLM	Log ANSI Labeled Tape Utility
SJ.AVOL2	OJ.AVOL2	AVOLM	Log ANSI Labeled Tape Utility
SJ.COMP1	OJ.COMP1	COMPRESS	Object Module Concatenation Utility
SJ.CRYPT	OJ.CRYPT	N/A	ANSI Data Encryption Routine
SJ.DECMP	OJ.DECMP	N/A	Compressed File Read Subroutine

Nonresident Routines (Cont.)

Source File Name	Object File Name	Load Module	Description
SJ.DEVL	OJ.DEVL	DEVINITL	Write Control Storage (WCS) Initializer
SJ.DSCMP	OJ.DSCMP	J.DSCMP	Disk Compress Module
SJ.DTSAVE	OJ.DTSAVE	J.DTSAVE	Auto Date and Time Update for Mounted Volumes
SJ.ENABLE	OJ.ENABLE	ENABLE	Terminal Online Task (SA only)
SJ.ERR	OJ.ERR	M.ERR	MPX-32 Abort Code Module
SJ.xx.ER	N/A	N/A	Sample Source for Error File
SJ.FORMF	OJ.FORMF	J.FORMF	Format Floppy Formatter Program
SJ.FREAD	OJ.FREAD	N/A	Read Subroutine for Key and Project
SJ.HELP	OJ.HELP	HELP	Help Service
SJ.HELPT	OJ.HELPT	HELPT	Help Translator
SJ.HLP	OJ.HLP	J.HLP	Help Load Module
SJ.INIT	OJ.INIT	J.INIT	System Initializer
SJ.KEY	OJ.KEY	KEY	M.KEY File Editor
SJ.KEYWD	OJ.KEYWD	KEYWORD	User Key Task
SJ.LABEL	OJ.LABEL	J.LABEL	Label ANSI Tape Utility
SJ.LIST	OJ.LIST	LIST	List File Utility
SJ.LOGCNT	OJ.LOGCNT	LOGCNT	Terminal Logon Counter Task
SJ.LOGTIME	OJ.LOGTIME	LOGTIME	Terminal Logon Timer Task
SJ.MDREST	OJ.MDREST	J.MDREST	Memory Disk Restore Task
SJ.MDSAVE	OJ.MDSAVE	J.MDSAVE	Memory Disk Save Task
SJ.MDTI	OJ.MDTI	J.MDTI	MDT Initialization Task
SJ.MOUNT	OJ.MOUNT	J.MOUNT	System Mount Service
SJ.OPCOM	OJ.OPCOM	OPCOM	Operator Communications
SJ.PAUSE	OJ.PAUSE	PAUSE	Pause Task
SJ.PROJ	OJ.PROJ	J.PRJCT	Project Accounting Utility
SJ.PSWD	OJ.PSWD	PASSWORD	User Password Task
SJ.REST	OJ.REST	RESTART	On-line Restart
SJ.SHAD	OJ.SHAD	J.SHAD	Shadow Memory RRS Utility
SJ.SOEX	OJ.SOEX	J.SOEX	Output Spooling Executive
SJ.SOUT	OJ.SOUT	J.SOUT	Output Spooler
SJ.SSIN	OJ.SSIN	J.SSIN1	Input Spooling - Files
SJ.SSIN	OJ.SSIN	J.SSIN2	Input Spooling - Devices
SJ.SWAPR1	OJ.SWAPR1	J.SWAPR	Nonresident Swapper (code)
SJ.SWAPR2	OJ.SWAPR2	J.SWAPR	Nonresident Swapper (parameters)
SJ.TDEFI	OJ.TDEFI	J.TDEFI	Terminal Definition Initialization Task
SJ.TERMOUT	OJ.TERMOUT	TERMOUT	Terminal Session Timer Task

Nonresident Routines (Cont.)

Source File Name	Object File Name	Load Module	Description
SJ.TINIT	OJ.TINIT	J.TINIT	Terminal Initializer
SJ.TSET	OJ.TSET	J.TSET	Set Terminal Type Utility
SJ.TSM	OJ.TSM	J.TSM	Terminal Service Monitor
SJ.UNLCK	OJ.UNLCK	J.UNLOCK	Dual Port Unlock Utility
SJ.UFDPT	OJ.UPDPT	N/A	Volume Formatter Disk Parameter Table
SJ.VFMT	OJ.VFMT	J.VFMT	Volume Formatter
SJ.VPRE	N/A	N/A	Volume Formatter and Volume Manager PRE File
SORT.MERGE	OJ.SORT.MERGE	FSORT2	Sort/Merge *
FUP**	N/A**	VOLMGR	Volume Manager **

* Directory SORT.MERGE contains the source file, SOR.SRC, used to create the object used to build the sort/merge library and FSORT2.

** Directory FUP contains all source files used to create the object library FUP.LIB and object directory FUP.DIR. These are used for building VOLMGR.

SYSGEN

The following are SYSGEN load module components:

Source File Name	Object File Name	Description
SJ.FMTIO	OJ.FMTIO	SYSGEN Formatter
SJ.OBUTL	OJ.OBUTL	Object Processor
SJ.PSCAN	OJ.PSCAN	SYSGEN Scanner
SJ.SDBUG	OJ.SDBUG	SYSGEN Debugger
SJ.SEXEC	OJ.SEXEC	Executive (Root Segment)
SJ.SGINI	OJ.SGINI	Initialization Overlay
SJ.SPH01	OJ.SPH01	Phase 1
SJ.SPH02	OJ.SPH02	Phase 2
SJ.SPH03	OJ.SPH03	Phase 3
SJ.SPH04	OJ.SPH04	Phase 4
SJ.SSCAN	OJ.SSCAN	Keyword Scanner
SJ.STACK	OJ.STACK	SYSGEN Stack
SJ.STBLS	OJ.STBLS	Device Type Table

The following are SYSGEN files:

File Name	Description
MSTRALL	System Image File for all CONCEPT 32/*xx* machines
MSTRALLS	System Symbol Table File for all CONCEPT 32/*xx* machines
MSTRALLD	Directive File for all CONCEPT 32/*xx* machines
MSTREXT	System Image File for extended-mode image
MSTREXTS	System Symbol Table File for extended-mode image
MSTREXTD	Directive File for extended-mode image
MSTROUT	Mapped Out System Image File for CONCEPT 32/2000
MSTROUTS	Mapped Out System Symbol Table File for CONCEPT 32/2000
MSTROUTD	Mapped Out Directive File for 32/2000
OH.32	Compressed System Object
OH.32_E	Extended Compressed System Object
OH.32_OUT	Mapped Out Compressed System Object
SG.32	Sample SYSGEN Macro/Directives
JH.32	COMPRESS Input Directives
JH.32_E	Extended COMPRESS Input Directives
JH.32_OUT	Mapped Out COMPRESS Input Directives

Miscellaneous

The following are Help text files:

File Name	Description
DS.HLP	Data Structures
GMPX.HLP	General MPX Help
PROC.HLP	Processors
SAT.HLP	System Administrator Tools
SVC.HLP	SVC Calls
TOP.HLP	Top Level Menu

The following are macro library files:

File Name	Description
M.EQUATESX32	X32 Macros for user data structures
M.MACLIB	MPX/RTM Macro Library
M.MPXMAC	MPX-32 Macro Library
M.OSEQUATESX32	X32 macros for system data structures
M.SERVICESX32	X32 macros for system services
SM.EQUATESX32	Source used to load M.EQUATESX32
SM.MPXMC	Source used to load M.MPXMAC (MPX-32 macros)
SM.OSEQUATESX32	Source used to load M.OSEQUATESX32
SM.RTMMC	Source used to load M.MACLIB (RTM compatible macro library)
SM.SERVICESX32	Source used to load M.SERVICESX32

The following are subroutine library files.

File Name	Description
MPXDIR	MPX-32 System Subroutine Directory (null)
MPXLIB	MPX-32 System Subroutine Library (null)
FUP.DIR	File Utility Program (VOLMGR) Subroutine Directory
FUP.LIB	File Utility Program Library
SORT.DIR	Sort/Merge Directory
SORT.LIB	Sort/Merge Library

The following are PRE files:

File Name	Description
MPX_EXT	PRE File for assembly of Extended MPX-32 modules
MPX_NON	PRE File for assembly of Nonextended MPX-32 modules
MPX_OUT	PRE File for assembly of mapped out modules (CONCEPT 32/2000 only)
MPXPRE	Default PRE File for macro assembler

The following are Job Control Language (JCL) files:

File Name	Description
JJ.A.HLP	JCL to assemble/catalog online help modules
JJ.A.NON	JCL to assemble/catalog nonresident modules
JJ.A.ONE	JCL to assemble a component of VOLMGR
JJ.A.RS1	JCL to assemble resident modules (Services/Processors)
JJ.A.RS2	JCL to assemble resident modules (Interrupts/Traps)
JJ.A.RS3	JCL to assemble resident modules (Device Handlers)
JJ.A.SGN	JCL to assemble/catalog SYSGEN
JJ.A.SWP	JCL to create swapper load module (J.SWAPR)
JJ.A.TDI	JCL to assemble/catalog J.TDEFI
JJ.A.VOL	JCL to batch all assemblies for components of VOLMGR
JJ.B.LIB	JCL to create/build the null MPXLIB/MPXDIR
JJ.B.MAC	JCL to create/build macro libraries M.MPXMAC,M.MACLIB
JJ.C.VOL	JCL to build VOLMGR
JJ.COMPR	JCL to create/load COMPRESSed files Object Module
JJ.ID	JCL to run SEARCHER to create MPX-32 ID (time/date and program ID information)
JJ.M.ERR	JCL to assemble/catalog M.ERR (MPX-32 Revision 3.*x* Aborts)
JJ.MSTR	JCL to create Master System Images
JJ.SORT	JCL to build Sort/Merge
JJ.XX.ER	JCL to assemble/catalog SJ.*xx*.ER (User Aborts)
BATCH.OS	JCL for building MPX-32 Rev. 3 by processing JCL files
MSTRSDT	JCL to build the Master SDT
MSTRSRCE	JCL to build the Master Source Tape

Support Files

The following are support files:

File Name	Description
LOGONFLE	Terminal initialization file
M.CNTRL	System Control File
M.KEY	Key file
M.PATCH	Patch file
MPX.PRO	Terminal Profile
MPX.PRO.NOTDEF	Terminal Profile when TERMDEF file is not installed
MPX.PRO.TDEF	Terminal Profile when TERMDEF file is installed
TDEFLIST	List of supported TERMDEF functions
TERMDEF	Terminal definition file
VOLM	Macro for invoking VOLMGR

Directories

FUP	File Utility Program containing source files for creating libraries used to build VOLMGR
OBJECT	Object files
OBJECT_E	Object modules for extended MPX-32
OBJECT_OUT	Object modules for mapped out (CONCEPT 32/2000 only)
PET	Performance Evaluation Tool
SORT.MERGE	Sort/Merge

2.3.1 Magnetic Tape

The format of the magnetic tape Master SDT is similiar to the User SDT. Both contain a tape boot loader followed by an MPX-32 image. The master tape, however, contains three MPX-32 images instead of one. The three images are:

- MSTRALL — default image for all CONCEPT 32/*xx* computers
- MSTREXT — extended-mode image
- MSTROUT — default image for CONCEPT 32/2000 computers

When building the Master SDT, the Volume Manager sets a flag enabling the tape boot loader to recognize a master boot and skip to the proper image for the appropriate CPU. If an image other than the default is desired, it can be selected using console panel mode commands. (See section 2.4, Booting a System from the Master SDT, in this chapter.)

2.3.2 Utility Tape

The utilities are contained on a separate tape, as an unbundled product, as shown in Figure 2-1. After the SDT has been restored, restore the desired utilities before exiting the Volume Manager.

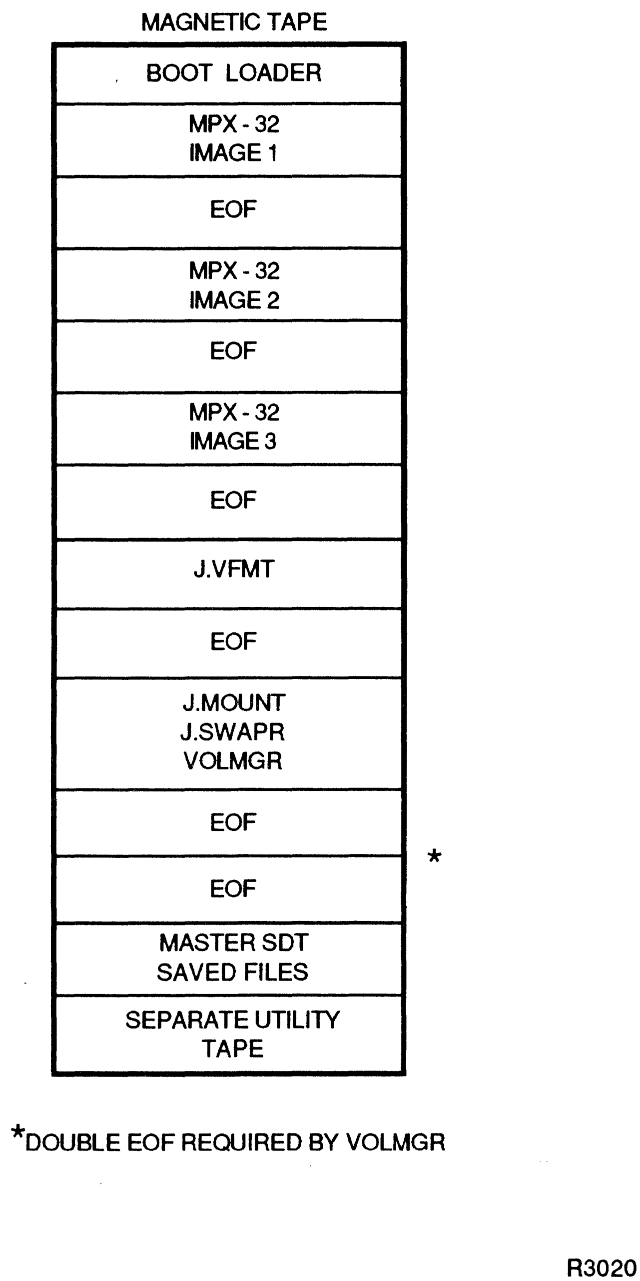

Figure 2-1
Master System Distribution Tape Format

2.4 Booting a System from the Master SDT

Mount the Master SDT on a tape drive and note its address. If the system console is connected to an IOP or MFP controller, the controller must be configured to channel X'7E'. If the system console is to be connected to the CONCEPT 32/2000 CPU's on-board F-class controller, it is recommended that the controller be configured to channel X'02'.

To boot a CONCEPT/32 computer with the system console connected to an IOP or MFP controller, perform the following steps on the system console:

Enter panel mode by typing:	`@@P`
System response:	`//`
Halt the system by typing:	`//HALT`
System response:	`//`
Reset the system by typing:	`//RST`
System response:	`//`
Clear memory by typing:	`//CLE`
System response:	`//`
Skip the next step if you wish to boot the default system image.	
Override the default system image by typing:	`//GPR0=`*n* *n* = 1 to select MSTRALL (non-extended MPX-32) image *n* = 2 to select MSTREXT (extended MPX-32) image
System response:	`//`
Load the system by typing:	`//IPL`=address of tape unit

(Skip the following instructions regarding booting on a windowing console and continue booting after the system has been loaded.)

To boot a CONCEPT 32/2000 computer with the system console connected to the F-class I/O channel on the CPU, perform the following steps on the windowing console:

Enter panel mode by activating the Panel Window on the windowing console.

System Response: The Panel Window drag bars become solid.

Halt the system, if not already halted, by selecting the `halt` command from the Panel Window menu.

System Response: The Node Status Window indicates that the CPU is halted.

Reset the system by selecting the `reset` command from the Panel Window menu.

System Response: The CPU registers and PSD displayed in the Panel Window are cleared.

Clear memory by selecting the `clear mem` command from the Panel Window menu.

System Response: The mouse pointer changes to `wait` while memory is being cleared.

Override the default system image by changing the contents of GPR0 to:

`gpr0:` `00000001` to select MSTRALL (non-extended MPX-32) image.

`gpr0:` `00000002` to select MSTREXT (extended MPX-32) image.

Load the system by selecting `ipl` from the Panel Window menu.

System Response: A sub-window prompting for the address of the tape unit appears.

Enter the tape unit's address in the sub-window.

The boot loader on the Master SDT begins execution. It selects and loads the proper system image for the CPU to be used, and passes control to SYSINIT, the system initialization program. The following prompts are displayed on the system console:

\>>

This prompt is displayed whenever the system debugger is configured in the system image being loaded. The system debugger is configured in all Master SDT images. Enter `TE` to continue normal system operation. When upgrading to a new revision of MPX-32, terminal initialization is inhibited at this point by responding to the system debugger as follows:

```
>> CM 780, 10000000
>> TE
```

This sets control switch 3.

`BOOT FROM A SCSI TAPE? (REPLY Y OR N):`

Enter `Y` if booting from a SCSI tape drive.

`MPX-32 MASTER SDT FOR CONCEPT 32/`*n* `COMPUTERS`

n is `67` for a 32/67, *n* is `87` for a 32/87, `97` for a 32/97, or `2000` for a 32/2000.

`ENTER SYSTEM DEVICE CHANNEL AND SUBADDRESS:`

Enter the channel and subaddress of the desired disk drive (for example, `0800`) as the logical device address where MPX-32 is to be installed. If an invalid system device address is entered, the following message appears:

```
CANNOT FIND PROPER DISC UDT...
SYSINIT ABORT RECEIVER ENTERED...
>>
```

```
ENTER DISC CONTROLLER TYPE: XIO (UDP OR DP II), HSDP, OR
MFP (REPLY X, H, OR M):
```

Enter the appropriate response for this disk prompt. If an incorrect value is entered, J.VFMT will detect the error during execution (see below) and abort.

```
MEMORY INITIALIZATION STARTED........
MEMORY INITIALIZATION COMPLETE.......
```

```
ENTER DATE AND TIME:
```

Enter the date and time using the following syntax:

date,*hh*[`:`]*mn*[`:`*ss*][`,` [`D`] [`,TZ=`*num*]]

date — is the current date in one of the following formats:

mm/dd/yy
dd-mm-yy
ddmmmyy

where *mm* is the 2-digit decimal month, *dd* is the 2-digit decimal day, *yy* is the 2-digit decimal year, and *mmm* is the 3-ASCII-character month abbreviation.

hh — is the 2-digit decimal hour (24 hour time)

mn — is the 2-digit decimal minute

ss — is the 2-digit decimal second. If the colon between *hh* and *mm* is omitted, this parameter is ignored.

`D` — indicates daylight savings time. Specifying this field causes the internally stored binary time to be adjusted by one hour.

`TZ=`*num* — allows the internal binary time to be biased by *num* hours. The value of *num* can be positive or negative. This field allows file times to be kept according to a given standard while the displayed time (see OPCOM TIME directive) is the correct local time.

Examples:

```
06/16/81,08:45:00
06/09/81,11:20:00
14-06-81,13:00:00
03NOV81,09:25:00
08/29/81,10:33,D
15-04-81,19:15:00,D,TZ=-3
05MAY81,16:15,,TZ=10
```

```
TASK LOADING FROM TAPE STARTED.....
```

`FMT>`

This is the Volume Formatter prompt. The Volume Formatter builds the on-disk structures required by the MPX-32 file system.

Important: See Chapter 13 of this volume for more information on the Volume Formatter. Many important user-selected parameters are only available at this point and cannot be changed later without removal of all user data on the disk.

If you are installing MPX-32 for the first time, and targeting a new disk media that has never been media verified, follow step A below. As an alternative to step A (INITIALIZE) you may choose to run the Level II Diagnostic Media Verification program (HSDP or UDP/DP II controllers) or the Level II Diagnostic program (MFP SCSI) before installing the SDT. The instructions in step A can also be used to reconfigure structures built by the INITIALIZE directive on a currently formatted disk.

If you are installing MPX-32 for the first time and targeting disk media that has been media verified, or if you are upgrading MPX-32, but wish to reconfigure currently formatted disk structures built by the FORMAT directive, follow step B below.

If you are only upgrading MPX-32 with a Master SDT tape, follow step C below.

Warning: Selecting steps A or B below will result in the removal of all user data on the disk. Be sure all important files have been saved to tape or another disk.

A. The minimum response required at this prompt is:

   `FMT> INITIALIZE DEVICE=`*devmnc* `DISC=`*dcode*

   devmnc is a 6-character device mnemonic (2-character device type mnemonic, 2-digit hexadecimal channel number, 2-digit hexadecimal device subaddress); for example, `DM0800` or `DM0402`.

   dcode *dcode* is the disk storage device type code. Refer to Note 1 in the INITIALIZE directive section of Chapter 13 for a list of disk type codes.

   If desired, other INITIALIZE command parameters may also be included on the command line. The INCLUDE parameter must not be used. If the FMAP parameter is specified, the line printer must be online.

   This command causes the hardware-required disk structures (e.g., track and sector labels) to be written to the disk. This is in preparation for item B below.

   At the completion of the INITIALIZE command, `FMT>` appears again. When this prompt appears, follow step B below.

B. The minimum response required at this prompt is:

`FMT> FORMAT DEVICE=`*devmnc* `VOLUME=`*volname*

devmnc is a 6-character device mnemonic (2-character device type mnemonic, 2-digit hexadecimal channel number, 2-digit hexadecimal device subaddress); for example, `DM0800` or `DF0802`.

volname is a 1- to 16-character volume name. Valid characters are A through Z, 0 to 9, dot (.) and underscore (_).

If desired, other FORMAT command parameters may also be included on the command line, for example, ACCESS=, CONFIRM=, MAXRES=. The IMAGE parameter must not be used. If the BOOTFILE parameter is used, the only acceptable values are `OLD` and `NEW`.

This command causes the volume structures required by the MPX-32 file system to be written to the disk, the selected image on the Master SDT to be the default image, and the disk bootstrap to be updated.

Volume Formatter will exit upon completion of this command.

C. Enter only the following at the Volume Formatter prompt:

`FMT> REPLACE DEVICE=`*devmnc* `VOLUME=`*volname*

devmnc is a 6-character device mnemonic (2-character device type mnemonic, 2-digit hexadecimal channel number, 2-digit hexadecimal device subaddress); for example, `DM0800` or `DM0402`.

volname is a 1- to 16-character volume name. Valid characters are A through Z, 0 to 9, dot (.) and underscore (_).

This command cause the image on the Master SDT to be the new default image and the disk bootstrap to be updated. Volume Formatter will exit upon succesfull completion of this command.

Note: If the Volume Formatter determines that volume structures exist on the specified disk device, this interactive prompt is issued:

`DESTROY OPTION NOT ENABLED FOR FORMATTED VOLUME,`
`VOLUME NAME=`*volname*
`CONTINUE - Y/N?`

where *volname* is the current volume name of the specified disk device.

Enter `Y` to continue. If you select `N`, you must rewind the SDT tape and start the boot procedure anew. Entering `DESTROY=Y` on the command line for steps A or B will prevent this prompt from appearing. This is not an option for step C.

For all steps (A, B, or C), if the `CONFIRM=N` parameter is not entered at the command line, an interactive prompt to continue is written to the screen. You may make changes in the displayed parameters or enter `CONFIRM=N` to continue.

Important: If any error messages are issued by Volume Formatter followed by an abort, take corrective action, rewind the SDT tape, and start the boot procedure anew.

`J.MOUNT - MOUNTING VOLUME` *volname* ON *devnmc*

If volume clean up is required, the following message is issued.

`J.MOUNT - VOLUME CLEAN UP ON` *volname*

`VOL>`

This is the Volume Manager prompt. When installing MPX-32 for the first time (e.g. steps A and B or step B above were performed), the system directory must be created before files on the SDT can be restored. Only the system directory is required. OBJECT, SOURCE, HELP, SORT.MERGE, BUILD.JCL, FUP, PET, REDUCE, MIPS.SRC, CSWI, INTR, and DEMO are optional and do not have to be created by the user.

The minimum response required at the Volume Manager prompt is:

`VOL> CREATE D SYSTEM ENTRIES=`*nn*

Note: The number of entries allowed in the SYSTEM directory (*nn* above) must be greater than 300 to accommodate all system files on the Master SDT.

If desired, other parameters such as OWNER=, PROJECTGROUP=, and ACCESS= may also be included with this directive.

Saved files on the Master SDT can now be restored. Restore the first three images using the RESTORE directive. Then exit Volume Manager.

`VOL> RESTORE VOLUME=SYSTEM` (information files and INSTALLSDT)

`VOL> RESTORE VOLUME=SYSTEM` (non-resident module)

`VOL> RESTORE VOLUME=SYSTEM` (non-resident module)

`VOL> EXIT`

A series of messages -- `DISC FFFF MARKED OFF LINE` -- are now displayed.

If J.TINIT is restored from the Master SDT, one of these initialization messages is displayed. The one that is displayed depends on whether the system file, LOGONFLE, exists or was restored on the currently mounted system volume. See Chapter 10 of this volume for details on how to build LOGONFLE.

```
INITIALIZATION COMPLETE
TERMINAL SETUP COMPLETE
```

(or)

```
INITIALIZATION COMPLETE

M.ASSN DENIAL, NO LOGON FILE, DEFAULT USED
TERMINAL SETUP COMPLETE
```

```
PRESS ATTENTION FOR TSM
```

Upon entering `@@A` for attention, a prompt is issued to enter a 1- to 8-character logon owner name and a 1- to 8-character logon key. The following characters cannot be used in owner names or keys: blanks, commas, semicolons, equal signs, line feeds, dollar signs, exclamation points, percent signs, and left or right parentheses. After entering a valid owner name and key, the TSM prompt is displayed and any valid TSM directive can be entered.

If a valid M.KEY file does not exist at logon, the following message is displayed prior to the `TSM>` prompt:

```
UNABLE TO ESTABLISH A DEFAULT WORKING DIRECTORY
YOUR CURRENT WORKING DIRECTORY IS @SYSTEM(SYSTEM)
```

Initiate the Master SDT tape installation macro by typing `INSTALLSDT`.

```
TSM> INSTALLSDT
```

This macro will restore the other images on the Master SDT. Through a series of interactive prompts, the macro allows you to choose which Master SDT images to restore, or simply restore all of them.

```
WOULD YOU LIKE A HARDCOPY OF THE SDT.CONTENTS FILE
TO ASSIST YOU IN THIS INSTALLATION ? (Y/N):
```

Enter `Y` to print a copy of the SDT.CONTENTS file.

```
IS THIS A COLD START (Y/N) :
```

For a cold start, enter `Y`. The following prompt will appear:

```
ENTER LINE PRINTER DEVICE ADDRESS (LP7EF8) :
```

Respond to this prompt by typing the address of the line printer device.

For a warm start, enter `N`.

The next prompt provides the option of restoring the entire Master SDT tape.

```
DO YOU WISH TO INSTALL THE ENTIRE TAPE ? (Y/N)
```

If you wish to restore every image on the Master SDT (except the first three), enter `Y` and follow step A below. If you wish to restore some images on the Master SDT, but not others, enter `N` and follow step B below.

A. After `Y` is entered, the following prompts appear.

   ```
   ENTER MAGNETIC TAPE DEVICE ADDRESS (M91000) :
   ```

   Enter the address at which the Master SDT is mounted.

ENTER THE VOLUME NAME WHICH THE SDT WILL BE INSTALLED ON :

Enter SYSTEM here.

MOUNT MESSAGES ARE NOW TURNED OFF UNTIL INSTALLATION HAS COMPLETED.

As each image is restored, a message appears stating that the image has been restored. After the process of restoring the Master SDT images is complete, the following appears:

THE SDT HAS BEEN INSTALLED ACCORDING TO THE INTSTRUCTIONS SUPPLIED BY THE USER.

The INSTALLSDT macro then rewinds the Master SDT tape, issues the following message, and exits:

MOUNT MESSAGES ARE NOW TURNED BACK ON

B. After N is entered, a series of prompts appears asking which images are to be restored. Each prompt lists a group of images and asks if these images should be restored. These prompts appear in the order shown below, grouping the images as follows:

- required images (nonresident routines, COMPRESS object, resident services and processor objects, resident interrupt and trap objects, resident device handler objects, HELP text files, macro libraries, PRE files, macros, TERMDEF, and SYSGEN files)
- extended MPX-32 image
- mapped-out images
- sort merge files
- unsupported load modules and source images
- patch file image
- optional files (can be skipped for warm starts)
- JCL files for building the operating system, source files contained on the SDT, and macro library source
- FUP library, nonresident object, and SYSGEN object
- Performance Evaluation Toolkit
- MIPSMON images, M.SURE test suite, and H.ICP test suite
- MPX-32 Demo images

If you enter Y in response to a prompt, all of the images listed in that prompt will be restored. If you enter N, none within that prompt will be restored.

After these 12 prompts have been responded to, follow step A above, begining with the ENTER MAGNETIC TAPE DEVICE ADDRESS prompt, to restore all those images responded to with Y.

After the INSTALLSDT macro has exited, mount any unbundled software tapes on the tape drive and restore them using the Volume Manage RESTORE directive.

Type the following to assign the task to tape:

```
TSM> AS TAP TO DEV=M9 BLO=N
```

Then type VOLMGR to re-enter Volume Manager. After all desired saved files are restored, type EXIT to exit the Volume Manager.

```
TSM> VOLMGR
VOL> RESTORE VOLUME=SYSTEM
      .
      .
      .
VOL> EXIT
```

Note: If the system does not install as described, check the hardware/firmware revisions of the system and contact the Encore Field Representative.

2.4.1 Control Switches

While rebooting the system, various initialization processes can be inhibited or enabled by setting the appropriate control switches. The assignment of the 13 switches is:

Switch	Function if set
0	inhibits volume clean-up by J.MOUNT
1	SYSINIT enters the system debugger before processing patches
2	inhibits patch processing (See Chapter 9, Entry Conditions.)
3	inhibits terminal initialization
4	inhibits accounting functions including the M.KEY, M.PRJCT, M.ACCNT, and M.ERR files
5	inhibits processing of the sequential task activation table at IPL time
6	If J.MOUNT encounters an invalid resource descriptor due to an invalid resource descriptor type field or space definition, it branches and links to the system debugger (if present) with R2 pointing to the resource descriptor.
7	J.MOUNT prereads the file space bit map (SMAP) or the resource descriptor allocation bit map (DMAP). J.MOUNT will not perform file overlap protection.
8	delete spooled output files instead of resubmitting them for processing
9	inhibits activating LOADACS during IPL or RESTART operations
10	enables faster memory initialization by checking only one location per map block to determine if that map block is present. It is not recommended that this switch be set on the first IPL after power up.
11	inhibits initialization of the memory descriptor table (MDT)
12	for RMSS: inhibits booting of nodes while J.BOOT executes

The control switches can be accessed by the console. The proper time to set the switches is while the system is waiting for the date and time to be entered. To set, for example, switch 3, the following must be entered at an MPF or IOP console:

```
ENTER DATE AND TIME: @@P
//CS=10000000 Terminal Initialization Inhibited
//@@C
<CR>
INVALID DATE FORMAT=MM/DD/XX
ENTER DATE AND TIME:
```

(Refer to the CONCEPT 32/2000 Operation manual for instructions on the windowing console.)

During power up, control switches are prezeroed if the proper firmware revision level has been installed. Power up without prezeroing can cause unexpected system responses due to incorrect control settings.

All control switch settings are preserved during system reboots not involving system power up (i.e., online restart and IPL).

2.5 Philosophy of Bootstrapping

The SDT process attempts to eliminate as much stand alone I/O from the startup code as possible because it is difficult to maintain stand alone capability for a number of different types of controllers. This philosophy is reflected in the operation of the system initialization program SYSINIT. For further details on SYSINIT, see section 2.6 of this chapter.

Because there may be no information on the disk at IPL time, tasks cannot be activated from the disk. Therefore, the four needed tasks (Volume Formatter, Mount, J.SWAPR, and Volume Manager) are supplied in load module format on the Master SDT. SYSINIT performs tape activations on these four load modules to allow the System Builder to:

- format the disk and build a bootable operating system (Volume Formatter)
- mount the system volume (Mount)
- activate the swapper (J.SWAPR)
- restore the files needed from the Master SDT and the Utility tape to make the system function (Volume Manager).

2.6 The System Builder

Following the operating system in memory is a two-part module called SYSINIT. This module is included by SYSGEN and is responsible for initializing the MPX-32 environment.

The first part, Phase I, runs stand alone. That is, MPX-32 is not yet functioning. Phase I loads the CPU scratchpad, enables interrupt levels, resets I/O channels, and performs any other cleanup operations required. It then sets up Phase II to run as an MPX-32 task. When Phase I enables interrupts, a clock interrupt occurs and causes Phase II to begin execution as a task.

Phase II activates the Volume Formatter, allowing the new volume to be formatted. Next, the volume mount service is activated and mounts the system volume. After the system volume is mounted, J.SWAPR is activated. Because there may be no information on the disk at IPL time, activations are made from the SDT, causing backward and forward movement of the tape. After this, the Volume Manager is run. This utility restores any files that are required for system operation. These files must include all key load modules and may also include any desired files. When Phase II completes, the system is fully initialized and must be opened for user access. Phase II accomplishes this by activating J.INIT, J.TINIT and J.TSM.

The user interaction for the above procedure on a Master SDT is described in section 2.4, Booting a System from the Master SDT.

The user interaction for the above procedure on a User SDT is described in Chapter 4 of this volume.

2.7 Operating Under the Starter System

The MPX-32 system is both disk and memory resident. TSM can be entered by entering `@@A` then the system owner name and no key. Valid owner names are created by the KEY utility.

2.8 Restoring Utility Processors, Libraries, and Other Files

The Volume Formatter (J.VFMT) and Volume Manager (VOLMGR) utilities are supplied on the Master SDT. All other utilities such as the Text Editor, Macro Assembler, and Media are an unbundled product. After all required files have been restored from the Master SDT, all desired unbundled software must be restored.

Load modules on the tapes are saved by using Volume Manager SAVE directives and can be restored from the tapes by using Volume Manager RESTORE directives. Each save directive allows up to 48 entries. An end-of-file (EOF) is written after each group saved.

When restoring files, use a restore directive for each group. Initially, all files should be restored from the Master SDT and the Utility tape using the Volume Manager RESTORE VOLUME directive. In subsequent interactions, selective restore capabilities of the Volume Manager can be used for system installation and maintenance.

The number of groups of files saved on the tapes can vary, and an equivalent number of restore directives are required to install them. A listing is provided with the Master SDT showing the groups in which the files were saved. Before continuing, check the listing. Restore all groups of libraries, system files, and utilities (one restore directive per group) using the listing as a guide.

2.9 Example of First Use of Master SDT on User System

The following examples show how to boot a starter system from the Master SDT. The first illustrates the entire booting process. The next two show the commands needed to choose a system image other than the default.

2.9.1 Using the Default Master SDT Image on a CONCEPT 32/97

```
     //HALT
     //RST
     //CLE
     //IPL=1000
>>TE
BOOT FROM A SCSI TAPE? (REPLY Y OR N): N
MPX-32 MASTER SDT FOR CONCEPT 32/97 COMPUTERS
ENTER SYSTEM DEVICE CHANNEL AND SUBADDRESS: 0800
ENTER DISC CONTROLLER TYPE: XIO (UDP OR DP II), HSDP, OR MFP
 (REPLY X, H, OR M): X
 MEMORY INITIALIZATION STARTED.....
 MEMORY INITIALIZATION COMPLETE.....
ENTER DATE AND TIME: 04/23/90,13:30
TASK LOADING FROM TAPE STARTED....
FMT> FORMAT DEVICE=DM0800 VOLUME=JONES MAXRES=3000-
FMT_CONFIRM=Y
      .
      .
      .
```

Enter CONFIRM=N to continue execution, or enter any changes.

```
FMT> CONFIRM=N
   DEVICE=DM0800 -- VOLUME FORMATTING SUCCESSFULLY
   COMPLETED
 J.MOUNT - MOUNTING VOLUME JONES ON DM0800
VOL> CREATE D SYSTEM ENTR=1000
VOL> RESTORE VOLUME=SYSTEM
VOL> RESTORE VOLUME=SYSTEM
VOL> RESTORE VOLUME=SYSTEM
VOL> EXIT
 DISC FFFF MARKED OFF LINE
 INITIALIZATION COMPLETE
 TERMINAL SETUP COMPLETE
 PRESS ATTENTION FOR TSM
@@A
TSM> INSTALLSDT
     WOULD YOU LIKE A HARD COPY OF THE SDT.CONTENTS FILE
     TO ASSIST YOU IN THIS INSTALLATION ? (Y/N) Y
     IS THIS A COLD START (Y/N) : Y
     ENTER LINE PRINTER DEVICE ADDRESS (LP7EF8) : LP7EF8
     DO YOU WISH TO INSTALL THE ENTIRE TAPE ? (Y/N) Y
     ENTER MAGNETIC TAPE DEVICE ADDRESS (M91000) : M91000
```

```
        ENTER THE VOLUME NAME WHICH THE SDT WILL BE
        INSTALLED ON: SYSTEM
        MOUNT MESSAGES ARE NOW TURNED OFF UNTIL INSTALLATION
        HAS COMPLETED.
         IMAGES 2 & 3 - NONRESIDENT ROUTINES ARE NOW
         BEING...
         IMAGE 4 - COMPRESS OBJECTS ARE NOW BEING...
         IMAGE 5 - RESIDENT SERVICES & PROCESSORS ARE NOW
         BEING...
           .
           .
           .
        THE SDT HAS BEEN INSTALLED ACCORDING TO THE
        INSTRUCTIONS SUPPLIED BY THE USER.
        MOUNT MESSAGES ARE NOW TURNED BACK ON
```

(The following steps install optional tapes.)

```
TSM> AS TAP TO DEV=M9 BLO=N
TSM> VOLMGR
VOL> RESTORE VOLUME=SYSTEM
  .
  .
VOL> EXIT
```

2.9.2 Overriding the Default Image

2.9.2.1 Selecting MSTREXT for CONCEPT 32/97

To select MSTREXT, the extended MPX-32 Master SDT image, enter the following:

```
     //HALT
     //RST
     //CLE
     //GPR0=2
     //IPL=1000
>>TE
BOOT FROM A SCSI TAPE? (REPLY Y OR N): N
MPX-32 MASTER SDT FOR CONCEPT 32/97 COMPUTERS
         .
         .
         .
```

(Continue as in the first example.)

2.9.2.2 Selecting MSTRALL for CONCEPT 32/2000

To select MSTRALL as the Master SDT image when booting a CONCEPT 32/2000 computer, do the following:

Enter panel mode by activating the Panel Window on the windowing console.

Halt the system, if not already halted, by selecting the `halt` command from the Panel Window menu.

Reset the system by selecting the `reset` command from the Panel Window menu.

Clear memory by selecting the `clear mem` command from the Panel Window menu.

Override the default system image by changing the contents of GPR0 to `gpr0: 00000001`.

Load the system by selecting the `ipl` command from the Panel Window menu.

Enter the tape unit's address in the sub-window.

```
>>TE
BOOT FROM A SCSI TAPE? (REPLY Y OR N): N
MPX-32 MASTER SDT FOR CONCEPT 32/2000 COMPUTERS
         .
         .
         .
```

(Continue as in the first example.)

2.9.2.3 Errors

Errors that occur while the user is selecting the Master SDT image cause a program halt. After such an error occurs, GPR4 and GPR5 contain the ASCII values for the message `BOOTFAIL` and GPR6 and GPR7 contain the ASCII values for one of the following error messages:

- `INV GPR0` — invalid value in GPR0 before IPL (must be 0, 1, or 2)
- `MACHTYPE` — invalid machine type from CPU status word. Valid machine types:

  0 = CONCEPT 32/2000
  3 = CONCEPT 32/67
  4 = CONCEPT 32/87
  5 = CONCEPT 32/97

- `INV IMAG` — invalid image selected (mismatch of image type and machine type)
- `LOAD MOD` — bad load module (mismatch of current and expected block number in reading image)
- `CHECKSUM` — checksum error in reading image
- `IO ERROR` — SDT tape I/O error

3 Building and Testing a System

3.1 Introduction

This chapter describes how to SYSGEN and test a resident system configured for a specific installation. Once the starter system has been installed, modifications can be made to the input files used by the System Generation utility, SYSGEN, to configure a tailored system. The modified data is saved to tape. This tape then becomes the user SDT. The SYSGEN utility is described in Chapter 7 of this volume. This chapter establishes SYSGEN in the cycle of building an MPX-32 system.

3.2 Building the SYSGEN Input Files

3.2.1 Building the Directive Input File

The directive input file for SYSGEN determines the configuration of the system. The input file specifies such items as: hardware to support, interrupts to connect, and devices to use for spooled system I/O. File SG.32, provided on the Master SDT, can be used as a model or as a working base for preparing a SYSGEN directive file for the CONCEPT/32. Either edit this file or build a directive input file from scratch using the Text Editor. The resulting file must be uncompressed.

The system debugger, DEBUG, can be included in the resident system by specifying a SYSGEN PROGRAM or USERPROG directive. This is recommended, particularly in an initial system. The system debugger adds approximately 300W to the size of the resident system. See Chapter 8 of this volume for further details on the system debugger.

The size of the target system image can be reduced by one 2KW map block by removing compatibility mode system modules H.ALOC, H.FISE, H.MONS, and H.CALM using the SYSGEN NOCMS directive. Because the system image end address is map block bounded, and the compatibility mode system modules total less than one map block, the size of the system image may not be affected if the compatibility mode modules are removed.

3.2.2 Building the Object Input File

Using the directive input file, SYSGEN determines which system modules, user modules, interrupt handlers, and trap handlers are needed to build the target system. SYSGEN reads the object code for these modules and handlers from the object input file. An object input file, OH.32, for the CONCEPT/32 is provided on the Master SDT.

User object modules can be added to the object input file. The names of these modules are added to the file JH.32 supplied on the Master SDT. The file is then compressed using the COMPRESS task. This task is described in the next section.

3.2.3 The COMPRESS Task

The COMPRESS task builds a file containing any number of object files. Its input file must contain the ASCII names, one per record/line, of the object modules to copy to the output file. The logical file code for input to COMPRESS is IN. The default assignment of IN is to the system file JH.32. The logical file code for output from COMPRESS is OT. The default assignment of OT is to the system file OH.32.

COMPRESS writes a control stream naming the files copied and lists the number of records each contains. It also reports any allocation or read errors.

Syntax

COMPRESS

The provided object input file used by SYSGEN is created using the COMPRESS task. The COMPRESS input file for the SYSGEN object input file is included on the Master SDT and can be a model or a working base for modifying the SYSGEN object input file. Proper execution of SYSGEN depends on the order of the following modules in the COMPRESS input file:

Oli H.IP06 must be specified before H.CALM is specified.

- H.IOCS must be specified before H.MVMT or H.BKDM is specified.
- H.XIOS must be specified before H.IFXIO is specified.
- H.IFXIO must be specified before H.??XIO is specified, where ?? are any valid ASCII characters.
- H.MUX0 must be specified before H.??MP is specified, where ?? are any valid ASCII characters.
- H.SWAPR must be specified before H.SINIT is specified. H.SWAPR and H.SINIT must be the last two names in the input file, if the system debugger is not to be configured in the system image. When the system debugger is specified (USERPROG=DEBUG) the unmapped portion, H.DBUG2, must be the last name in the input file immediately following H.SINIT. The mapped portion of the system debugger, H.DBUG1, can be positioned anywhere in the input file before H.SWAPR.
- H.MUX0 and H.IP00 must both reside in the first 64K of memory.
- H.EXSUB must be specified before H.EXEC, H.EXEC2, or H.EXEC3 is specified.

H.IOCS, H.IFXIO, and H.MUX0 are only specified one time in the input file.

COMPRESS IN Assignment	COMPRESS OT Assignment/ SYSGEN OBJ Assignment
JH.32	OH.32

Example

```
TSM> CHAN DIRE=^OBJECT
TSM> ASSIGN  IN TO JH.32 BLOCKED=Y
TSM> ASSIGN  OT TO OH.32 BLOCKED=Y
TSM> ASSIGN  LO TO LFC=UT
TSM> COMPRESS
COPIED - 2 RECORDS FROM PATH pathname
COPIED - 2 RECORDS FROM PATH pathname
                  .
                  .
                  .
TSM>
```

pathname is the name of the file from which the records were copied.

If a complete pathname is specified in the COMPRESS input file, the complete pathname is displayed. If only a file name is specified in the COMPRESS input file and the file is found on the current working volume and directory, only the file name is displayed. If only a file name is specified in the COMPRESS input file and the file is not found on the current working volume and directory, the system volume and directory are searched; if the file is found, the complete pathname is displayed. If a file name specified in the COMPRESS input file cannot be found, a message is displayed and the COMPRESS task continues execution.

3.3 Running SYSGEN

For information on interactive and batch access, required and default assignments, and other aspects of running SYSGEN refer to Chapter 7 of this volume. This section describes a simple path for configuring a system.

The logical file code (LFC) for a SYSGEN directive file is DIR. A modified version of SG.32 created by the user (called `SG.DIR` in the example below) can be assigned to DIR.

The object input file, OH.32, can be assigned to OBJ, which is the logical file code for the SYSGEN object input file. A TSM ASSIGN directive can assign the input files as shown below.

```
TSM> ASSIGN OBJ TO OH.32 BLOCKED=Y
TSM> ASSIGN DIR TO SG.DIR BLOCKED=Y
TSM> SYSGEN
```

Alternatively, a job file can run SYSGEN in the batch mode with the above file assignments.

The SYSGEN output file to contain the system image is specified with the SYSTEM directive in the SYSGEN directives file. The output file is then created by SYSGEN. File space for the output file need not be created before running SYSGEN. The file name specified with the SYSTEM directive is the name to specify as the system load file when restarting to test a system image.

3.4 Testing a SYSGENed System

After the tailored image is configured, it can be tested by restarting with the RESTART directive or the shutdown task, J.SHUTD. Any owner can use the RESTART directive unless prohibited in the M.KEY file. (Refer to Chapter 5 of this volume for more information on RESTART.) The J.SHUTD task is available only to the system administrator. (Refer to Chapter 10 of this volume for more information on J.SHUTD.)

Restarting without specifying an image file or performing IPL from the console causes the default system image (the starter system when running from SDT) to return as the working system image. You can continue developing configurations by using the editor, SYSGEN, and one-shot restarts as needed. Figure 3-1 shows the online restart test process in overview.

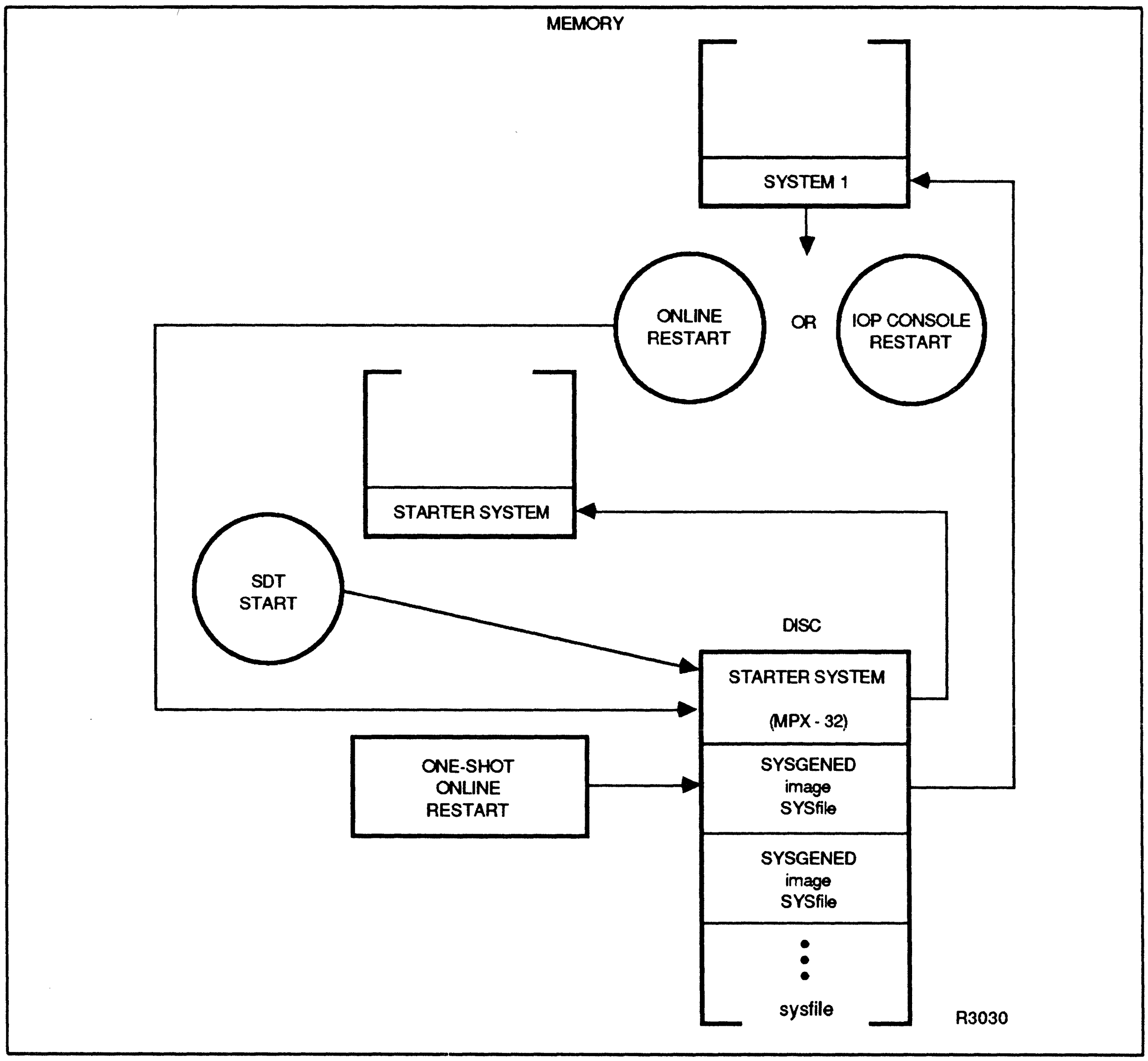

Figure 3-1
Testing a User-Configured System

3.5 Terminal Initialization and System Protection

Chapter 10 of this volume describes how to initialize terminals on MPX-32 and how to build an M.KEY file to authorize owners for access to the system and implement owner privileges. If LOGONFLE and M.KEY files exist, RESTART initializes terminals and M.KEY privileges.

4 Installing a User-Configured System

4.1 Introduction

Once a configured system is tested and ready to install in place of the starter system, the Volume Manager builds a user System Distribution Tape (SDT).

4.2 Creating a User System Distribution Tape (SDT)

A user SDT is created by the Volume Manager and is similar to the Master SDT. Figure 4-1 shows the layout of a user SDT. The first image on the SDT, called the SDT image, contains the desired system image and four essential load modules that must be configured in the following order for the system to be built properly:

- Volume Formatter (J.VFMT)
- System Mount (J.MOUNT)
- Swapper (J.SWAPR)
- Volume Manager (VOLMGR)

See MPX-32 Reference Manual, Volume II, Chapter 3 for information on creating the SDT.

During SDT processing, the load modules on the SDT image are loaded into memory and automatically activated. They are not restored to the system disk since directory information has not been retained about them. Therefore, these four modules must also be saved in one of the save images following the SDT image. At the VOLMGR prompt during SDT processing, these load modules can then be restored to the system disk.

In addition to the system image and essential load modules, another group of files required for system operation must be saved: OPCOM, J.INIT, and J.TSM. These files must also be restored to disk when the VOLMGR program is activated. Therefore, the minimum required modules on the SDT save are: OPCOM, VOLMGR, J.INIT, J.MOUNT, J.TSM, J.SWAPR, and J.VFMT.

The recommended set of load modules for an SDT is: VOLMGR, J.MOUNT, J.VFMT, J.SWAPR, OPCOM, J.INIT, J.TSM, J.SOUT, J.SSIN1, J.SSIN2, J.TINIT, J.SOEX, and EDIT.

As boot programs cannot process multivolume headers, a user SDT should not be created on multivolume tape.

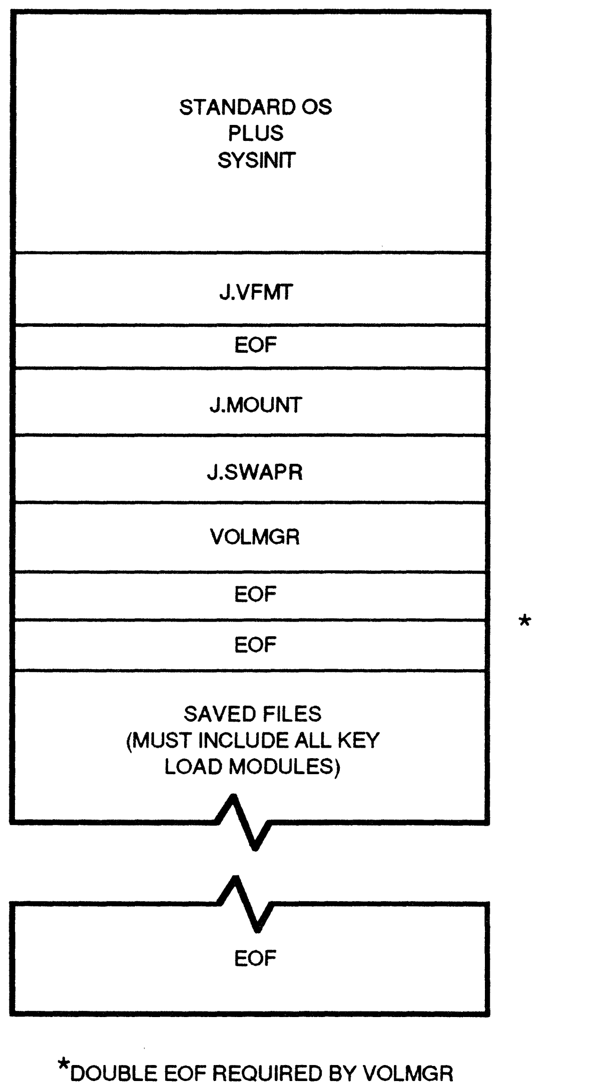

Figure 4-1
User System Distribution Tape Format

4.3 Installing a User System Distribution Tape (SDT)

All devices specified in the SYSGEN configuration should be connected.

The procedure to install a user SDT is similar to the procedure to install a Master SDT as described in Chapter 2 of this volume.

J.TINIT searches for a file in the system directory named LOGONFLE. The LOGONFLE can be built using the editor as documented in Chapter 10 of this volume. If LOGONFLE exists as a system file, J.TINIT uses the parameters contained in the file to initialize the terminals on the system. When terminal initialization is completed, the following message is displayed on the console:

```
INITIALIZATION COMPLETE
TERMINAL SETUP COMPLETE
SYSTEM READY...  PRESS ATTENTION FOR TSM
```

If LOGONFLE does not exist as a system file, J.TINIT uses default parameters for all terminals configured in the current system image (default parameter values are described in Chapter 2 of this volume). Once terminal initialization is completed and default values are assumed, the following message is displayed on the console:

```
INITIALIZATION COMPLETE
M.ASSN DENIAL, NO LOGON FILE, DEFAULT USED
TERMINAL SETUP COMPLETE
SYSTEM READY...  PRESS ATTENTION FOR TSM
```

4.4 Saving/Restoring System Processor and Utility Load Modules

Additional load modules, libraries, and files are saved and restored using Volume Manager save and restore directives.

4.5 Booting a System from a User SDT

To boot a CONCEPT/32 computer with the system console connected to an IOP or MFP controller, perform the following steps on the system console:

Enter panel mode by typing:	`@@P`
System response:	`//`
Halt the system by typing:	`//HALT`
System response:	`//`
Reset the system by typing:	`//RST`
System response:	`//`
Clear memory by typing:	`//CLE`
System response:	`//`
Load the system by typing:	`//IPL`=address of tape unit

The following prompts are displayed on the system console:

This prompt is displayed only if the system debugger is configured in the system:

```
>>
```

Enter `TE` to continue normal system operation.

```
ENTER SYSTEM DEVICE CHANNEL AND SUBADDRESS:
```

Enter the 2-character channel number and 2-character subaddress of the system volume, for example, `0800` or `0801`.

```
MEMORY INITIALIZATION STARTED.....
MEMORY INITIALIZATION COMPLETE.....
```

```
ENTER DATE AND TIME:
```

Enter the date and time using the following syntax:

date,hh[`:`]*mn*[`:`*ss*] [`,` [`D`] [`,TZ=`*num*]]

date is the current date in one of the following formats:

mm/dd/yy
dd-mm-yy
ddmmmyy

mm is the 2-digit decimal month, *dd* is the 2-digit decimal day, *yy* is the 2-digit decimal year, and *mmm* is the 3-ASCII-character month abbreviation.

hh is the 2-digit decimal hour in 24 hour time.

mn is the 2-digit decimal minute.

ss is the 2-digit decimal second. If the colon between *hh* and *mn* is omitted, this parameter is ignored.

`D` indicates daylight savings time is in effect. Specifying this field causes the internally stored binary time to be adjusted by one hour.

`TZ=`*num* allows the internal binary time to be biased by *num* hours. The value of *num* can be positive or negative. This field allows file times to be kept according to a given standard while the displayed time by the OPCOM TIME directive is the correct local time.

Examples:

```
06/16/81,08:45:00
06/09/81,11:20:00
14-06-81,13:00:00
03NOV81,09:25:00
08/29/81,10:33,D
15-04-81,19:15:00,D,TZ=-3
05MAY81,16:15,,TZ=10
```

```
TASK LOADING FROM TAPE STARTED.....
```

Note: If an invalid system volume channel or subaddress was entered before entering the date and time, the following message appears at this point:

```
INVALID SYSTEM OR SWAP VOLUME ENTERED
```

The user must then enter a valid system volume channel and subaddress. Booting then continues with the FMT> prompt as show below.

```
FMT>
```

This is the Volume Formatter prompt. The Volume Formatter builds the on-disk structures required by the MPX-32 system.

Important: See Chapter 13 of this volume for more information on the Volume Formatter. Many important user-selected parameters are only available at this point and cannot be changed later without removal of all user data on the disk.

For successful installation of the current release of MPX-32, the minimum response to this prompt is the REPLACE directive (step C).

If you are installing MPX-32 for the first time, and targeting a new disk drive that has never been media verified, follow step A below. As an alternative to step A (INITIALIZE) you may choose to run the Level II Diagnostic Media Verification program (HSDP or UDP/DP II controllers) or the Level II Diagnostic program (MFP SCSI) before installing the SDT. The instructions in step A can also be used to reconfigure structures built by the INITIALIZE directive on a currently formatted disk.

If you are upgrading MPX-32, but wish to reconfigure currently formatted disk structures built by the FORMAT directive, follow step B below.

If you are only upgrading MPX-32 with a user SDT tape, follow step C below.

Warning: Selecting steps A or B below will result in the removal of all user data on the disk. Be sure all important files have been saved to tape or another disk.

A. The minimum response required at this prompt is:

`FMT> INITIALIZE DEVICE=`*devmnc* `DISC=`*dcode*

devmnc is a 6-character device mnemonic (2-character device type mnemonic, 2-digit hexadecimal channel number, 2-digit hexadecimal device subaddress); for example, `DM0800` or `DM0802`.

dcode *dcode* is the disk storage device type code. Refer to Note 1 in the INITIALIZE directive section of Chapter 13 for a list of disk type codes.

If desired, other INITIALIZE command parameters may also be included on the command line. The INCLUDE parameter must not be used. If the FMAP parameter is specified, the line printer must be online.

At the completion of the INITIALIZE command, `FMT>` appears again. When this prompt appears, follow step B below.

B. The minimum response required at this prompt is:

`FMT> FORMAT DEVICE=`*devmnc* `VOLUME=`*volname*

devmnc is a 6-character device mnemonic (2-character device type mnemonic, 2-digit hexadecimal channel number, 2-digit hexadecimal device subaddress); for example, `DM0800` or `DM0802`.

volname is a 1- to 16-character volume name. Valid characters are A through Z, 0 to 9, dot (.) and underscore (_).

If desired, other FORMAT command parameters may also be included on the command line, for example, ACCESS=, CONFIRM=, MAXRES=. The IMAGE parameter must not be used. If the BOOTFILE parameter is used, the only acceptable values are `OLD` and `NEW`. Volume Formatter will exit upon completion of this command.

C. Enter only the following at the Volume Formatter prompt:

`FMT> REPLACE DEVICE=`*devmnc* `VOLUME=`*volname*

devmnc is a 6-character device mnemonic (2-character device type mnemonic, 2-digit hexadecimal channel number, 2-digit hexadecimal device subaddress); for example, `DM0800` or `DM0402`.

volname is a 1- to 16-character volume name. Valid characters are A through Z, 0 to 9, dot (.) and underscore (_).

This causes the image on the user SDT to be the new default image. Volume Formatter will exit upon completion of this command.

Note: If the Volume Formatter determines that volume structures exist on the specified disk device, this interactive prompt is issued:

```
DESTROY OPTION NOT ENABLED FOR FORMATTING VOLUME,
VOLUME NAME=volname
CONTINUE - Y/N?
```

Enter `Y` to continue. If you select `N`, you must rewind the SDT tape and start the boot procedure anew. Entering `DESTROY=Y` on the command line for steps A or B will prevent this prompt from appearing. This is not an option for step C.

For all steps (A, B, or C), if the `CONFIRM=N` parameter is not entered at the command line, an interactive prompt to continue is written to the screen. You may make changes in the dispalyed parameters or enter `CONFIRM=N` to continue.

If any error messages are issued by Volume Formatter, take corrective action, rewind the SDT tape, and start the boot procedure anew.

```
J.MOUNT - MOUNTING VOLUME JONES ON DM0800
```

```
VOL>
```

This is the Volume Manager prompt. If a system directory larger than the default is required, a system directory can be created before files on the SDT are restored. However, if the REPLACE command was used above (meaning the volume had already been formatted), the system directory may already exist on the disk. If the system directory already exists, do not execute the CREATE DIRECTORY command.

```
VOL> CREATE D SYSTEM ENTRIES=xx
```

Other parameters can also be included on the directive line, for example OWNER=, PROJECTGROUP=, ACCESS=.

Saved files on the SDT can now be restored. Each RESTORE directive processes one save image. Repeat this directive until all desired saved files on the SDT have been restored. Mount any unbundled software tapes on the tape drive and restore in the same manner. After all desired saved files are restored, type EXIT to exit the Volume Manager.

```
VOL> RESTORE VOLUME=SYSTEM
              .
              .
              .
VOL> EXIT
```

If using floppy disks, only the system image is on the boot floppy. Saved files are on additional floppy disks.

```
INITIALIZATION COMPLETE
TERMINAL SETUP COMPLETE
```

(or)

```
INITIALIZATION COMPLETE
M.ALOC1 DENIAL, NO LOGON FILE, DEFAULT USED
TERMINAL SETUP COMPLETE
```

If J.TINIT is restored from the SDT, one of these initialization messages is displayed; which one depends on whether the system file, LOGONFLE, exists on the currently mounted system volume. See Chapter 10 of this volume for details on how to build LOGONFLE.

```
PRESS ATTENTION FOR TSM
```

The system is now fully operational. Upon entering `@@A` for attention, a prompt is issued to enter a 1- to 8-character logon owner name and 1- to 8-character logon key. The following characters cannot be used in owner names or keys: blanks, commas, semicolons, equal signs, line feeds, dollar signs, exclamation points, percent signs, and left or right parentheses. After entering a valid owner name and key, the TSM prompt is displayed and any valid TSM directive can be entered.

4.5.1 Example

```
//HALT
//RST
//IPL=1000

>>TE
ENTER SYSTEM DEVICE CHANNEL AND SUBADDRESS:0800
 MEMORY INITIALIZATION STARTED.....
 MEMORY INITIALIZATION COMPLETE.....
ENTER DATE AND TIME: 27APR90,08:00:00
TASK LOADING FROM TAPE STARTED.....
FMT> FORMAT DEVICE=DM0800 VOLUME=JONES MAXRES=3000-
FMT_CONFIRM=Y
                 .
                 .
                 .
```

Enter CONFIRM=N to continue exection, or enter any changes.

```
FMT> CONFIRM=N
   DEVICE=DM0800 -- VOLUME FORMATTING SUCCESSFULLY
   COMPLETED
   J.MOUNT - MOUNTING VOLUME JONES ON DM0800
VOL> CREATE D SYSTEM ENTR=1000
VOL> RESTORE VOLUME=SYSTEM
                 .
                 .
                 .
VOL> EXIT
PRESS ATTENTION FOR TSM
```

5 Online Restart

5.1 Introduction

An online restart halts the current system and restarts to the default or specified image. It does not alter control switch settings. Online restart makes separate allocation requests to allocate the volume descriptor (block 4) and the bootstrap blocks (blocks 0-3) of the target disk.

Online restart is initiated with the TSM $RESTART directive or by specifying the RESTART option for the J.SHUTD task. Refer to Chapter 10 of this volume for more information on J.SHUTD.

5.2 Bootstrap Programs

MPX-32 has two standard bootstrap progams. The first bootstrap program (referred to as OLD in the Volume Formatter chapter of this volume) loads images requiring no more than five Input/Output Command Doublewords (IOCDs). The second bootstrap program (called NEW) handles images requiring six or more IOCDs to load, as well as those requiring no more than five.

The volume descriptor reserves space for five Read IOCDs. This is sufficient for loading images up to 50 KB (hexadecimal), but it leaves no space for additional IOCDs. To accommodate images requiring more than five IOCDs, the NEW bootstrap program itself contains the image load (Read) IOCDs.

Online restart supports both bootstrap programs. However, when restarting to a disk containing an OLD-style bootstrap program, an image requiring six or more IOCDs cannot be used as the restart image.

To determine the bootstrap program being used on a volume, perform a disk dump (DD) of the disk's volume descriptor (block 4). If the old bootstrap program is being used, VO.IOCD1 contains read IOCDs. If the new bootstrap program is being used, VO.IOCD1 contains the following ASCII text:

```
NEW DISC BOOTSTRAP. IOCDS IN BOOT BLOCK.
```

Note: Because of current firmware limitations, MFP SCSI disks may not be able to use the NEW standard bootstrap program or NEW-style custom bootstrap programs. MFP SCSI images that require six or more IOCDs to load must use MFP ECO 88-188, Revision E or greater. A six-IOCD image can be restarted, but cannot be made the default image and IPLed when using less than Revision E. Refer to MFP ECO 88-188 for more information about the MFP solutions and availability.

5.3 TSM $RESTART Directive

The TSM $RESTART directive is available to any user unless restricted in the M.KEY file. This directive can be used to:

- restart the current default image or specified image
- one-shot test a new system image
- establish a new default system image

Figure 5-1 illustrates how a default system image is established.

Note: Because this directive does not re-initialize the Analog/Digital Interface (ADI) board or its associated Real-Time Peripheral (RTP) Equipment, system images containing the ADI handler should be brought up using IPL procedures.

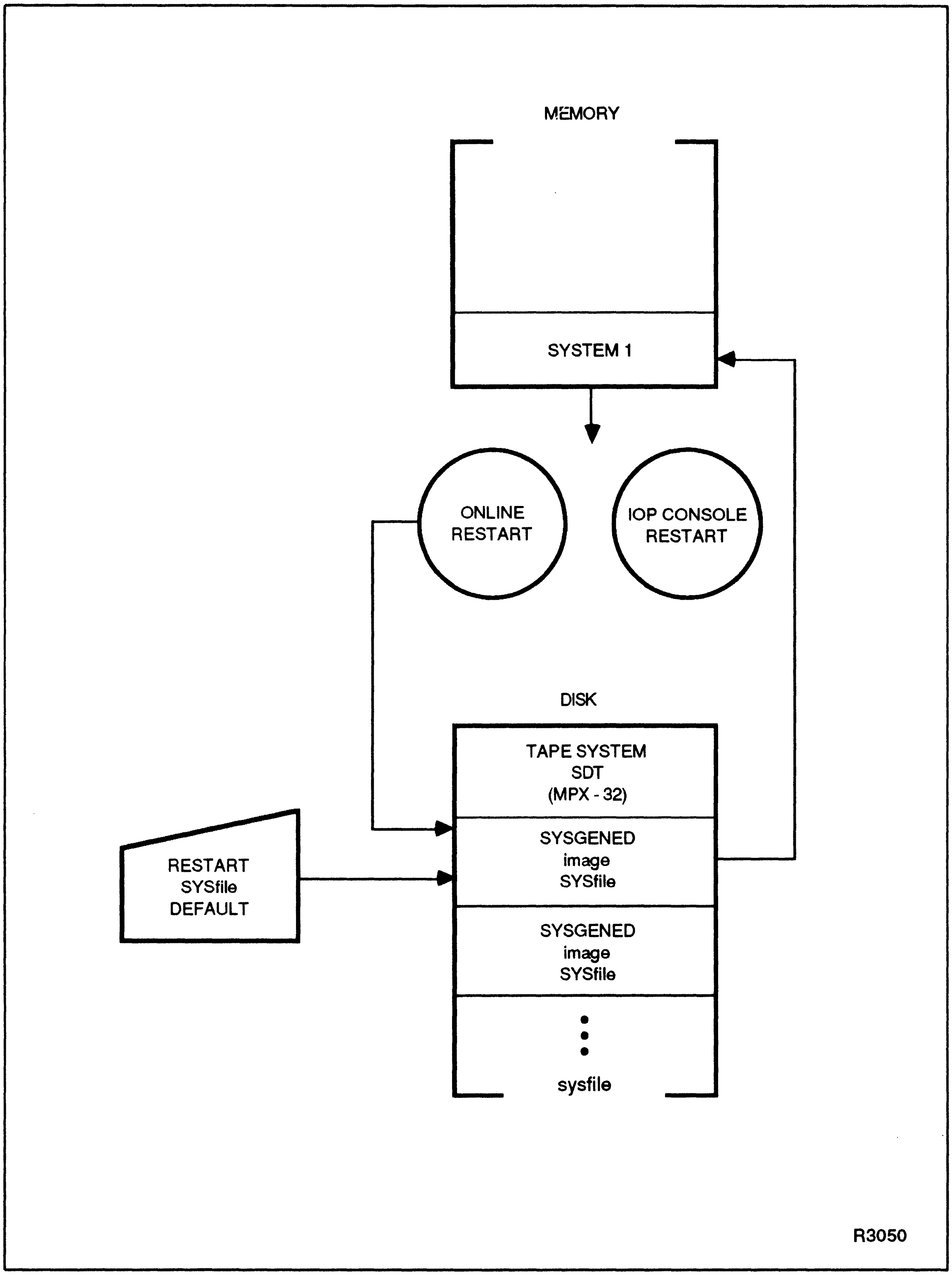

Figure 5-1
Establishing a New Default System

5.4 Precautions

Before using the TSM RESTART directive, perform the following steps:

- Use the TSM $SIGNAL directive to notify all terminal users to stop interactive and batch activity and logoff.
- Use the OPCOM LIST directive to see if any user tasks remain active. Abort any tasks by task number using the OPCOM ABORT directive.
- Check batch stream activity. It should end as users logoff.
- Because independent tasks are killed and pending I/O is lost during restart, reactive or re-establish independent tasks following a restart.

5.5 Using $RESTART

Syntax

$RESTART [*sysfile*]

sysfile is the pathname of the file containing the image to restart. This filename is the same as the filename supplied with the file's SYSGEN SYSTEM directive. If the filename is omitted, $RESTART uses the current default image.

$RESTART issuses the following prompts:

```
DO YOU WISH TO RESTART? (Y OR N):
DO YOU WISH THIS TO BE YOUR DEFAULT IMAGE? (Y OR N):
```

Note: Do not specify an image as the default image until it has been thoroughly tested.

Possible responses to these prompts and their resulting actions are:

Restart Prompt	Default Image Prompt	Resulting Action
Y	Y	restarts and establishes a new system default image
	N	restarts to specified image, but does not change default image (one-shot test)
N	Y	no restart, but establishes new default image
	N	no action. MPX-32 ignores the RESTART directive.

To one-shot test a new image, restart the image but respond `N` to the default image prompt.

5.6 Errors

If an allocation request for the volume descriptor or the bootstrap blocks fails, RESTART displays one of the following messages and ends processing:

```
RESTART...UNABLE TO ALLOCATE VOLUME DESCRIPTOR

RESTART...UNABLE TO ALLOCATE BOOTSTRAP BLOCKS
```

If the bootstrap program on a disk cannot be identified or properly handled (e.g. it is a custom bootstrap), RESTART issues the following messages and ends processing:

```
RESTART...INVALID BOOTSTRAP ON DISC VOLUME

RESTART...EXECUTE J.VFMT "REPLACE" TO WRITE NEW
BOOTSTRAP AND IMAGE TO VOLUME
```

When using the old bootstrap program, if the user restarts to an image requiring six IOCDs to load and specifies that image as the default, RESTART displays the following message but does not end processing:

```
RESTART...NEW IMAGE TOO LARGE TO MAKE DEFAULT WITH
CURRENT BOOTSTRAP ON DISC
```

Subsequent IPLs result in loading of the previous image.

When using the old bootstrap program, if the user restarts to an image requiring seven or more IOCDs to load and specifies that image as the default, RESTART displays the following messages and ends processing:

```
RESTART...NEW IMAGE TOO LARGE TO RESTART WITH CURRENT
BOOTSTRAP ON DISC

RESTART...EXECUTE J.VFMT "REPLACE" TO WRITE NEW
BOOTSTRAP AND IMAGE TO VOLUME
```

If the user restarts to the default image when no system volume is configured, RESTART aborts with the following message:

```
RESTART...RESTART TO DEFAULT IMAGE CANNOT BE PERFORMED
ON REMOTE NODE
```

6 Recovering the System

6.1 Introduction

If the operating system halts, a fresh copy of the disk file containing the operating system image can usually be restored into memory from the IOP console. This is an initial program load (IPL) and restart operation, and goes through the restart cycle illustrated in the previous chapter.

The following instructions apply to input on an IOP/MFP console. When using a windowing console (CONCEPT 32/2000 only), refer to the CONCEPT 32/2000 Operations Manual.

6.2 Recovery from Disk at the Console

To IPL and restart the default MPX-32 system from disk, enter the following at the console:

```
//RST
//IPL xxxx
```

xxxx is the address of the system disk.

Restart logic is always located on the system disk, and directs processing to the most recent default system file regardless of its disk location.

6.3 Errors During Start-up

If the system halts at or near location X'6FC' during IPL from disk, enter `GPR` at the // prompt to read the contents of R1.

R1 contains one of the following error codes that will aid in debugging:

Error Code	Description
1	checksum error detected on system image
2	system image was not located at address specified to bootstrap
3	fatal I/O error reading system image
4	error retry attempts exhausted
5	system does not match machine type

If the system halts during system initialization (SYSINIT) Phase I, R5 contains abort code SY01. R1 contains one of the following error codes:

Error Code	Description
1	unrecognizable IPL device indicator
2	invalid machine type
3	no UDT associated with the IPL device
4	target system image not compatible with CPU model
5	cannot boot extended mode MPX-32 on a 32/27
6	a device is configured at the CPU's F-class channel, preventing H.SINIT from dynamically configuring an operator console for a 32/2000 system
7	no priorities are available for the dynamically configured operator console on a 32/2000 system

If the system halts during SYSINIT Phase II, an abort message is displayed on the system console.

6.4 System Halt Analysis

If the operating system halts, the following procedure can be used on a CONCEPT/32 computer to determine what operations were in progress when a halt occurred.

1. Check the CPU front panel to determine if the halt was caused by a hardware failure. Check interrupt active, run, halt, and any other pertinent indicators.

2. Read PSW and Instruction:

   ```
   //PSW
   ```

3. Read registers 0 - 7:

   a. `//GPR`

   b. If R7=X'54524150', a trap, the registers contain the following information:

Register	Contents
0	PSD word 1
1	PSD word 2
2	real address of the instruction causing the trap
3	instruction causing the trap
4	trap status word
5	ASCII trap type (e.g. MF01)
6	address of registers saved at time of trap

If R5 = X'564D3939' (a potential file overlap) and R7 = X'00000034' (a double allocation error) occur, the registers contain the following information:

R0 = PSW
R1 = VOMM FCB address
R2 = MVTE address
R3 = requested size in number of allocation units
R4 = next available SMAP bit position
R6 = internal VOMM error code

If R5 = X'564D3939' (a potential file overlap) and R7 = X'00000000' (a double deallocation error) occur, the registers contain the following information:

R0 = PSW
R1 = VOMM FCB address
R2 = MVTE address
R3 = MVTE address
R4 = start SMAP bit position to perform deallocation
R6 = SMAP start block number

c. The registers at the time of the trap are examined by entering:

`//MA = x`

x is the contents of R6

followed by seven returns. Each return displays the contents of the next register.

d. PSD2, real address, instruction, and status is examined by entering four returns, one return for each item.

4. Access system debugger, if configured:

a. `//RST`	
b. `//MA=B5C`	C.DEBUG (address of system debugger or 0 if not included in SYSGEN)
c. `//PC=x`	*x* is the address displayed as `MD=x` resulting from the `MA=B5C` step
d. `//RUN`	

5. In the system debugger:

a. `DT`	systems with event trace enabled
b. `ABS`	
c. `ECHO`	
d. `REMAP`	
e. `8E8`	determines current task
f. `8E8,+40`	dumps dispatch queue information
g. `DU 0,nnnn`	*nnnn* is the address of H.IP00, obtained from the SYSGEN load map
h. `DU N,2000N`	user task TSA

6. Make a dump tape, if a crash dump routine is configured
   a. Mount a tape on MT1000
   b. `//HALT`
   c. `//RST`
   d. `//MA=C28` — C.CRDUMP (address of crash dump routine or 0 if H.DEBUG2 or H.DMPMT is not included in system)
   e. `//PC=`*x* — *x* is the address displayed as `MD=`*x* resulting from the `MA=C28` step
   f. `//RUN`
   g. When the system is up, use ANALYZE to study the crash dump

6.5 Automatic IPL

If automatic IPL is hardware enabled and the system halts because of a power failure, the CPU initiates an IPL during the power up initialization. Automatic restart, the ability to restart or continue a software program that was interrupted by a power outage, is not supported. The software does, however, provide the required parameters for user implementation of the power up automatic restart feature.

If operator intervention is inhibited, all prompts normally displayed on the system console during system initialization are inhibited. The system provides default responses for the inhibited prompts. See the Operator Intervention Inhibit section, in Chapter 10 of this volume.

7 System Generation (SYSGEN)

7.1 Introduction

System Generation (SYSGEN) for an MPX-32 system involves supplying a set of configuration directives to the SYSGEN utility. Using these directives, the utility creates a permanent file containing the installation specific MPX-32 system in memory image absolute format.

7.2 General Description

SYSGEN is a privileged system utility that operates in a standard MPX-32 system. SYSGEN can be executed in batch or interactive mode. The system where SYSGEN is executed must have enough free memory to hold the generated system and SYSGEN itself. SYSGEN requires 16KW.

The resources that SYSGEN requires are:

- directives
- system object modules

The directives can be supplied interactively or by using batch from magnetic tape, disk, or card reader. If supplied on a magnetic tape, the tape can include SYSGEN resident object modules and MPX-32 modules. The file for the resident system image and the system table file are permanent disk files created by SYSGEN.

System object modules include: interrupt and trap processors such as H.IP00 and H.IPIT, modules that form the MPX-32 nucleus such as H.EXEC and H.REMM, resident system tasks such as J.SWAPR and the system debugger and the object for SYSINIT. These modules are provided as files to be restored from the Master SDT after a starter system is installed. The file naming convention for SDT files that are designed to be part of the resident system is OH.*module*.

User object modules for interrupt handlers, resident system tasks, and user-callable modules defined with the MODULE directive must also be included in the object file for SYSGEN.

A task called COMPRESS, provided on the Master SDT, concatenates object code. This task selects all required system module files for SYSGEN into the SYSGEN object input file. The COMPRESS input file can be modified to contain any modules, interrupt handlers, resident tasks, or device handlers needed to configure the resident operating system. See Chapter 3 of this volume for more information on COMPRESS.

Any handlers or tasks that are SYSGENed must be Assembler object modules that conform to the following basic structure. The modules must begin with a handler address table (HAT), end with an initialization entry point, and use the following system macros: M.EIR, M.XIR, M.MODT, M.SVCT, and M.SVCP.

System tables are constructed and linked to the resident system modules, handlers, and user-supplied resident modules and handlers as specified by SYSGEN directives. A resident system image is formed and subsequently written to the dynamically acquired disk file specified in the SYSTEM directive. Concurrent with this process, a listing of directives is built and a load map of the system is generated. The symbol table can be saved in a system symbol table file specified with the SYMTAB directive and used in patching the system.

SYSGEN uses big blocking buffers for object processing. To inhibit use of these blocking buffers, set option 1 before activation.

7.3 SYSGEN Logical File Codes

The logical file codes associated with SYSGEN are directives file (DIR), object module file (OBJ), base object module file (OBR), and listed output (SLO).

DIR Default and Optional Assignments

The file containing SYSGEN directives is assigned to logical file code DIR. The default assignment for DIR is to SYC. The optional assignment for DIR is to *pathname*.

`$ASSIGN DIR TO` *pathname* `BLO=Y`

pathname is the name of a file containing the SYSGEN directives

OBJ Default and Optional Assignments

The file containing system resident modules is assigned to logical file code OBJ. There are two default assignments for OBJ: OH.32_OUT (mapped out) is the default if the system image has been specified for a CONCEPT 32/2000 (see the MACHINE directive in this chapter); OH.32 is the default if it has not. In addition, there is an optional assignment for OBJ:

`$ASSIGN OBJ TO` *pathname* `BLO=Y`

pathname is the name of a file containing system resident object files

OBR Default and Optional Assignments

The file containing extended system modules is assigned to logical file code OBR.

OBR must not be reassigned when building a mapped out image.

The default for OBR is to NU (null device). There is one optional assignment for OBR:

`$ASSIGN OBR TO` *pathname* `BLO=Y`

pathname is the name of the file containing extended system modules

SLO Default and Optional Assignments

SYSGEN listed output — the directives list, the load map, and the error list are assigned to logical file code SLO.

The default assignment for SLO is to SLO.

There are three optional assignments for SLO:

`$ASSIGN SLO TO {DEV=`*devmnc* | `LFC=UT` | *pathname*`}`

`DEV=`*devmnc*
: *devmnc* is the device mnemonic of a device to contain SYSGEN listed output

`LFC=UT`
: specifies the logical file code UT

pathname
: is the name of the file to contain SYSGEN listed output.

7.3.1 LFC Summary

LFC	Default Assignment	Optional Assignment
DIR	SYC	*pathname*
OBJ	OH.32 (or OH.32_OUT)	*pathname*
OBR	NU (null device)	*pathname*
SLO	SLO	*pathname* DEV=*devmnc* LFC=UT

7.4 Options

Option	Description
1	big blocking buffers inhibited
17	each overlay module is debuggable
18	enter SYSGEN debugger when initializing system modules
19	enter SYSGEN debugger after a break or abort request is issued.

Options 17, 18, and 19 are processed only when bit C.DEV of C.BIT is set. This bit must be set to debug SYSGEN.

7.5 Accessing SYSGEN

SYSGEN is accessed from batch or TSM in this way:

[$] [EXECUTE] SYSGEN

or

$RUN @SYSTEM(SYSTEM)SYSGEN

When directive input (DIR) is assigned to a terminal, a `SYS>` prompt is displayed.

To exit SYSGEN and return to TSM, use <ctrl> C.

7.6 SYSGEN Directives

There are three types of SYSGEN directives:

- Section directives — begin with // and indicate the beginning of the three major sections: //HARDWARE, //SOFTWARE, and //END.
- Subsection directives — begin with / and indicate the subsections within major sections.
- Keyword directives — have no slash, and are part of a subsection.

SYSGEN directives begin in column 1 of the record and contain no embedded blanks. An asterisk in column 1 specifies a comment line. Within a line, a blank starts comment processing.

Numeric values are represented by decimal numbers unless otherwise specified.

Directives, except TITLE, can be continued across more than one input line by placing a hyphen (-) as the last significant character on the line to be continued.

SYSGEN directives are optional unless specifically described as required. In general, the order or presentation of SYSGEN directives is not critical within a directive subsection. However, if the order is critical, the individual directive discussion clarifies the proper order.

File SG.32 contains a starter SYSGEN directive file. It can be used as it exists or it can be modified. It contains the required directives and a wide assortment of possible configurations that can be activated by removing the comment designation (*).

SYSGEN directives are summarized below in the order they could appear in your directive file. Directives are described individually, in alphabetical order, on the following pages.

Directive	Description
TITLE	permits identifying information to be printed on the listed output file
//HARDWARE	indicates the beginning of the hardware section of directives
/PARAMETERS	designates the parameters subsection of the //HARDWARE section

Directive	Description
MACHINE	specifies the type of computer for which the system is being configured
IPU	specifies an Internal Processing Unit will be configured into the target system
/MEMORY	designates the memory subsection of the //HARDWARE section
RLWU	specifies whether read/lock write/unlock applies to all memory
SIZE	specifies the memory configuration of the target system
/CHANNELS	designates the controller and device directives subsection of the //HARDWARE section
CONTROLLER	specifies hardware channels to be configured
DEVICE	defines configured hardware I/O devices
/INTERRUPTS	designates the interrupt directives subsection of the //HARDWARE section
PRIORITY	specifies the interrupt configuration and the interrupt processors to be used by the target system
/TRAPS	designates the trap directives subsection of the //HARDWARE section
PROGRAM	specifies the names of trap handlers to be configured on the system
SYSTRAP	specifies the default trap handlers that are to be overridden. This directive can only be used if a PROGRAM directive is not specified.
USERPROG	specifies the names of trap handlers and resident system modules to be configured on the system if a PROGRAM directive is not specified
/SYSDEVS	designates the system device directives subsection of the //HARDWARE section
LOD	specifies a system listed output device. Used as the default device in related OPCOM directives
POD	specifies a system punched output device. Used as the default device in related OPCOM directives.
SID	specifies a system input device (SID). Used as the default device in related OPCOM directives.
SWAPDEV	specifies the default swap device
SWP	This directive is ignored.

Directive	Description
/VP	designates the Vector Processor subsection of the //HARDWARE section
VP	specifies the device characteristics of the Vector Processors to be configured
VPID	specifies the unit-specific information for a Vector Processor
//SOFTWARE	indicates the beginning of the software section of directives
/PARAMETERS	designates the parameters subsection of the //SOFTWARE section
SYSTEM	specifies the name of the file to be used as storage for the generated MPX-32 resident image
SYMTAB	specifies the name of the file to be used as storage for the system symbol table
DELTA	selects optional IPU/CPU scheduling algorithm
BATCHMSG	specifies suppression of task messages sent to users' terminals or the system console
DISP	determines the number of entries in the dispatch queue.
LOGON	specifies single or multiple logons using the same owner name
POOL	specifies the size of the memory pool to reserve
IOQPOOL	specifies the number of words to reserve for the IOQ memory pool
MSGPOOL	specifies the number of words to reserve for the MSG memory pool
MTIM	specifies the number of real-time clock interrupts per second
NTIM	specifies the number of real-time clock interrupts per time unit
ITIM	specifies the time interval set for the interval timer on the RTOM module
NOTSMEXIT	specifies that TSM remains active when not in use
TSMEXIT	specifies that TSM exits from the system when no longer required
ITLB	generates an Indirectly Connected Task Linkage Block
MMSG	specifies the maximum number of no-wait messages to be sent by an unprivileged task
MRUN	specifies the maximum number of no-wait run request to be sent by a task
MNWI	specifies the maximum number of no-wait I/O requests that can be concurrently outstanding for a task

Directive	Description
TQFULL	specifies the largest time quantum a single user time-distribution task acquires before being relinked to the bottom of the priority list at its base execution priority
TQMIN	specifies the smallest time quantum a single user task acquires before preemption by a higher priority user time-distribution task
BATCHPRI	specifies the execution base priority for batch jobs
TERMPRI	specifies the execution priority level for all tasks activated in the interactive terminal environment
PATCH	specifies a patch area to append to the MPX-32 resident image
MODE	requests special system operations
SVC	increases the size of the SVC table
FLTSIZE	This directive is ignored.
RMTSIZE	increases the size of the Resourcemark Table
ACTIVATE	specifies the names of load modules to be activated after the system has been booted. Status is not checked.
SEQUENCE	specifies the names of load modules to be activated after the system has been booted. Status is checked.
TRACE	allows initialization of the system trace flag word C.TRACE
DEBUGTLC	allows specification of the console address for the System Debugger's stand-alone I/O
PCHFILE	provides the name of the file to be used as the patch file for the generated system
DBGFILE	provides the name of the file containing the default task debugger load module for the generated system
DPTIMO	specifies a default time-out value to be applied to resource delays encountered when attempting to access a multiprocessor, shared-volume resource
DPTRY	specifies the decimal number of tries to obtain a multiprocessor resource before issuing a denial
KTIMO	specifies the number of seconds the kill directive will attempt to abort a task before it kills it
DTSAVE	specifies the elapsed time before the date/time backup program resumes
SWAPSIZE	specifies the initial swap file size
SWAPLIM	specifies the minimum partial swap quantum
EXTDMPX	specifies the location for extended memory MPX-32 and/or the TSA for tasks cataloged SYSTSA

Directive	Description
HELP	specifies the directory containing help files and activates J.HLP
DEMAND	specifies which tasks are demand page processed
NODEMAND	inhibits demand page processing
AGE	specifies the amount of virtual time to pass before an unreferenced page is considered aged during demand page processing
BEGPGOUT	specifies the minimum percentage of total memory desired for free pages on a demand page system
ENDPGOUT	specifies the maximum percentage of total memory desired for free pages on a demand page system
MAPOUT	designates mapped out as the default execution mode for tasks cataloged with the ENVIRONMENT SYSMAP keyword
NOMAPOUT	designates mapped in as the default execution mode for tasks cataloged with the ENVIRONMENT SYSMAP keyword
/MODULES	designates the modules subsection of the //SOFTWARE section
MODULE	defines the name of an optional user module to be included in the MPX-32 resident image, the module number to be associated with the module, and the number of SVC callable entry points in that module
/OVERRIDE	designates the override subsection of the //SOFTWARE section
SYSMOD	replaces a system module with another module, or removes modules H.ALOC, H.CALM, H.FISE, and H.MONS
NOLACC	disables the MPX-32 functionality for tracking last date, time, and owner name of access to task-level resource
NOANSI	excludes support for ANSI labeled tapes from the system image
NOBASE	automatically excludes support for the execution of tasks utilizing base register addressing from the system image
NOCMS	automatically excludes support for MPX-32 Rev. 1 compatibility mode services (modules H.ALOC, H.FISE, H.MONS, and H.CALM) from the system image
CMIMM	returns the MPX-32 Revision 3.3 functionality for implicit volume mount management
CMPMM	inhibits public volume dismount

Directive	Description
NOTDEF	excludes support for the TERMDEF Facility from the system image
NOSYSVOL	inhibits mounting and use of system volume for RMSS systems
/PARTITION	designates the partition subsection of the //SOFTWARE section
NAME	defines Datapool or Global Common memory partitions
OWNER	specifies an owner name and access rights that apply to a memory partition defined by a NAME directive
PROJECT	specifies a project group name and access rights which apply to a memory partition defined by a NAME directive
OTHERS	specifies access rights that apply to users of a memory partition defined by a NAME directive who are not the owner or members of the associated project group
/SECURITY	specifies the security subsection of the //SOFTWARE section
OWNERNAME	inhibits echoing of owner name at logon
PASSWORD	specifies that passwords are required for logon
SAPASSWD	disables the PASSWORD task so that only the system administrator can change a password
SYSONLY	specifies SYSTEM as the only valid ownername if there is no M.KEY file
/TABLES	designates the tables subsection of the //SOFTWARE section
CDOTS	specifies the size of the user CDOT array
JOBS	specifies the maximum number of batch jobs that can be active concurrently
MDT	enables rapid file allocation through a memory resident descriptor table (MDT)
SHARE	specifies the number of entries in the shared memory table
TIMER	specifies the number of timer entries to be generated in the MPX-32 resident image
/RMSTABLS	designates the Resource Management System Tables subsection of the //SOFTWARE section
ARTSIZE	specifies the size of the allocated resource table (ART)
/FILES	designates the system files subsection of the //SOFTWARE section
SMD	This directive is ignored.

Directive	Description
SYCSIZE	included for compatibility only. This directive is accepted by SYSGEN, but the results are not used in job processing.
SGOSIZE	included for compatibility only. This directive is accepted by SYSGEN, but the results are not used in job processing.
//END	required as the last SYSGEN directive

7.6.1 ACTIVATE Directive

The ACTIVATE directive names load modules to be activated by SYSINIT immediately after the target system is booted. Status checks are not performed on these tasks. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

ACTIVATE=(*name1,...name7***)**

name1,...name7
: are the 1- to 8-character ASCII load module names to be activated, separated by commas. A maximum of 7 names can be entered per directive.

7.6.2 AGE Directive

The AGE directive indicates the amount of virtual time allowed to pass before an unreferenced page is considered aged during demand page processing. Virtual time is the time the task has control of the CPU.

If NODEMAND is specified, this directive is ignored. If this directive is not specified and demand page processing is supported, the default is the value in effect for TQMIN.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

AGE=*xx*

xx
: is the amount of virtual time in milliseconds (decimal) allowed to pass before an unreferenced page is considered aged. If not greater than zero, the value in effect for TQMIN is used.

7.6.3 ARTSIZE Directive

The ARTSIZE directive specifies the size of the allocated resource table (ART). The number of entries in the ART determines the maximum number of system resources that can be allocated in the system at one time. The size of each entry is 8 words. If this directive is not specified, 100 entries are reserved for the ART.

This directive appears under the //SOFTWARE /RMSTABLS subsection.

Syntax

ARTSIZE=*entries*

entries is the number of entries to be reserved for the allocated resource table (ART).

7.6.4 BATCHMSG Directive

The BATCHMSG directive inhibits batch messages from going to the system console, or to terminals logged on with the owner name to which the message is sent, or both. If the BATCHMSG directive is not specified, messages will go to all appropriate terminals and to the system console.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

BATCHMSG= ({ {TERM | NOTERM | CONS | NOCONS} | {TERM | NOTERM}, {CONS | NOCONS} })

TERM J.TSM allows batch messages to go to all terminals except the system console

NOTERM J.TSM inhibits batch messages from going to all terminals except the system console

CONS J.TSM allows batch messages to go to the system console

NOCONS J.TSM inhibits batch messages from going to the system console

If neither TERM or NOTERM is specified, TERM is the default. If neither CONS or NOCONS is specified, CONS is the default.

If both NOTERM and NOCONS are specified, task messages will not appear anywhere.

7.6.5 BATCHPRI Directive

The BATCHPRI directive specifies the execution priority level of all batch jobs. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is not specified, the priority level is 61.

Syntax

BATCHPRI=*nn*

nn is the 2-digit decimal time-distribution priority level, 55-64, to use for batch jobs.

7.6.6 BEGPGOUT Directive

The BEGPGOUT directive specifies the minimum percentage of total memory desired for free pages on a demand page system, the lower page-out threshold. When the BEGPGOUT value is reached, J.SWAPR begins writing modified, unreferenced pages which are queued for page-out to the swap file.

If NODEMAND is specified, this directive is ignored. If this directive is not specified and demand page processing is supported, the default is 10% of the total available memory on the system.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

BEGPGOUT=*xx*

xx is a percentage (00 through 100) of the total available memory on the system. H.SINIT converts this percentage to the correct number of pages (map blocks). *xx* must be less than the value specified for ENDPGOUT.

7.6.7 CDOTS Directive

The CDOTS directive specifies the size of the user CDOT array which follows the fixed communications area in lower memory. The user array can be read by any task but can only be written into by privileged tasks. C.USERVA is the starting point of the user array. If this directive is not specified, no user communications region is established.

If specified, this directive appears under the //SOFTWARE /TABLES subsection.

Syntax

CDOTS=*number*

number is the decimal number of words in the user communications variable array

7.6.8 /CHANNELS Directive

The /CHANNELS directive designates the controller and device subsection of the //HARDWARE section. This directive is required.

Syntax

/CHANNELS

7.6.9 CMIMM Directive

The CMIMM directive returns the MPX-32 3.3 revision functionality for implicit mount management. This includes the following features:

- OPCOM-mounted volumes can only be dismounted using the OPCOM DISMOUNT directive.
- TSM $CHANGE DIRECTORY directive allows access to any nonpublic user volume that is logically mounted to J.TSM.
- Logical dismount of the last user or assigner on a nonpublic user volume causes a physical dismount of that volume.

This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

CMIMM

7.6.10 CMPMM Directive

The CMPMM directive inhibits public volume dismount. This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

CMPMM

7.6.11 CONTROLLER Directive

The CONTROLLER directive represents one hardware channel to be configured in the generated system. This directive is required and must be repeated for each channel to be configured. The CONTROLLER directive appears under the //HARDWARE /CHANNELS subsection.

The HANDLER keyword permits specification of reentrancy and controller definition table (CDT) generation. The type of reentrancy specified the first time a handler name appears in a CONTROLLER or DEVICE statement is used for the entire system.

The CDT per unit definition table (UDT) specification applies until another HANDLER keyword or CONTROLLER statement is processed. Nonpresent channels can be configured.

Syntax

CONTROLLER=*ttcc*,**PRIORITY=***intlev*,**CLASS=***class* [,**CACHE**]
[,**HANDLER=**{*name* | (*name* [,I|,S] [,C])}] [,**MUX=***type*] [,**SUBCH=***a*]

tt is the 2-character ASCII device mnemonic (see Table 7-1)

cc is the 2-digit hexadecimal channel number

PRIORITY=*intlev*
intlev is the hexadecimal interrupt level. If the device class (see *class* below) is memory disk, a priority interrupt is not required. If the interrupt is on an IOP or MFP device, the priority must be the same as other devices on that IOP/MFP.

CLASS=*class*
class is the device class

D specifies 16MB addressable E class

F specifies extended I/O

M specifies memory disk

[,**CACHE**] specifies a cache controller. All devices under a cache controller are also cache.

[,**HANDLER=**{*name* | (*name* [,I|,S] [,C])}]

name is the 1- to 8-character handler name (see Table 7-1). If not specified, the following defaults are used: H.IFXIO (XIO, IOP, or MFP controllers), H.MUX0 (GPMC controllers), and H.NUXIO (null device).

I specifies interrupt priority level reentrancy, one copy per channel

S specifies system level reentrancy, one copy per system

C specifies one CDT for each UDT

[,**MUX=***type*]
type is the type of multiplex controller being configured:

GPMC specifies general purpose multiplex controller and includes H.GPMCS in the system image

XIO specifies extended I/O and includes H.XIOS in the system

IOP specifies input/output processor and includes H.XIOS in the system image

MFP specifies multi-function processor

[,SUBCH=*a*]

a is the subchannel the controller is connected to. This subchannel will be used to verify proper device address specifications on subsequent device directives. For example, the subchannel should match the first device address digit.

Table 7-1
MPX-32 Device Type Handlers

Device Mnemonic	Device Handler	Default Handlers	Device Description
CT	H.CTXIO	H.CTXIO	operator console (not assignable)
DC	H.DCXIO	H.DCXIO	any disk unit except memory disk
DM	H.DCXIO	H.DCXIO	any moving head disk
DM	H.MDXIO		any memory disk (HANDLER must be specified)
DM	H.DCSCI		any SCSI disk (HANDLER must be specified)
DF	H.DCXIO	H.DCXIO	any fixed head disk
MT	H.MTXIO	H.MTXIO	any magnetic tape unit
MT	H.MTSCI		any SCSI tape unit (HANDLER must be specified)
M9	H.MTXIO	H.MTXIO	any 9-track magnetic tape unit*
M7	H.MTXIO	H.MTXIO	any 7-track magnetic tape unit*
CR	H.CPMP		any card reader
LP	H.LPXIO	H.LPXIO	any line printer
PT	H.CPMP		any paper tape reader/punch
TY	H.F8XIO	H.F8XIO	any terminal/teletypewriter (other than console) **
CT	H.CTXIO	H.CTXIO	operator console (assignable)
FL	H.DCXIO	H.DCXIO	floppy disk
NU	H.NUXIO	H.NUXIO	null device
CA		N/A	communications adapter (binary synchronous/ asynchronous)
U0-U9		N/A	any available user-defined applications
LF		N/A	line printer/floppy controller (used only with SYSGEN)
ANY	H.DCXIO	H.DCXIO	any nonfloppy disk except memory disk

* When both 7- and 9-track magnetic tape units are configured, the designation must be 7-track.

** GPMC terminals have H.ASMP as their default handler.

Notes:

Multiple controller directives are invalid on a single channel, even if the devices configured on the channel have mixed device types.

Extended I/O handlers default to system reentrant handlers. The extended I/O interrupt fielder (H.IFXIO) is channel reentrant. One copy should be specified for each extended I/O channel configured by using the I parameter.

GPMC device handlers default to system reentrant handlers.

If a line printer and a floppy disk are configured on the same IOP channel, only one CONTROLLER directive should be used. Multiple DEVICE directives specify the device and handler.

The XIO common subroutines (H.XIOS) are not named within the SYSGEN directive file. They are automatically included during SYSGEN if MUX=XIO or MUX=IOP is specified.

The GPMC subroutines (H.GPMCS) are not named within the SYSGEN directive file. They are automatically included during SYSGEN if MUX=GPMC is specified.

Memory disks must be configured on channel 00 as DM00.

The High Speed Data Handler (HSDG) is a D-class I/O controller that must be SYSGENed as interrupt priority reentrant, one copy per channel, using the generic software handler, H.HSDG. An example of the controller and device statements is:

```
CONTROLLER=U040,PRIORITY=0A,CLASS=D,HANDLER=(H.HSDG,I)
DEVICE=00,DTC=U0
```

The SG.32 file on the SDT contains examples of how to SYSGEN other types of controllers.

7.6.12 DBGFILE Directive

The DBGFILE directive names the permanent file containing the default task debugger load module for the system being generated. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is not specified, the default filename is AIDDB.

Syntax

DBGFILE=*filename*

filename is the 1- to 8-character ASCII file name of the file containing the default task debugger load module

7.6.13 DEBUGTLC Directive

The DEBUGTLC directive specifies the console address for the System Debugger's stand-alone I/O. If this directive is not specified, the default address is the address of the system console. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

DEBUGTLC=*cc*

cc is a 2-digit hexadecimal channel number.

7.6.14 DELTA Directive

The DELTA directive selects the optional IPU/CPU scheduler. This directive appears under the //SOFTWARE /PARAMETERS subsection. When DELTA is specified, H.EXEC2 replaces H.EXEC, and H.CPU2 replaces H.CPU. For more information, see the Internal Processing Unit section, in Chapter 2 of MPX-32 Reference Manual Volume I.

Syntax

DELTA=*cc*

cc is a 2-digit decimal number between 0 and 54 specifying the IPU bias task boost value

Notes:

The IPU directive must be present to perform the substitution. If it is not specified, the DELTA directive is ignored.

Do not include H.CPU2 or H.EXEC2 in the USERPROG directive, as they are automatically selected.

7.6.15 DEMAND Directive

The DEMAND directive specifies which tasks are demand page processed.

When the mapped out object file is assigned at SYSGEN execution, demand page processing is supported, unless NODEMAND is specified. The DEMAND directive determines which eligible tasks are demand page processed. If other than the mapped out object file is assigned, demand page is not supported and this directive is ignored.

If this directive is not specified and demand page processing is supported, the following tasks are demand paged:

- eligible tasks at a priority range of 55 to 64
- tasks at priority levels 1 through 54 which are cataloged to run demand page

For more information about task eligibility for demand paging, refer to Reference Manual, Volume I, Chapter 3.

When specified, this directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

DEMAND={ NONE |*pp* }

NONE specifies that only those tasks specified as demand page tasks when cataloged or linked are demand page processed

pp is the highest priority level at which an eligible task runs in demand page mode. The default is 55.

7.6.16 DEVICE Directive

The DEVICE directive defines the configured hardware I/O devices. This directive is required. The DEVICE directive appears under the //HARDWARE /CHANNELS subsection.

The HANDLER keyword permits specification of reentrancy and of controller definition table (CDT) generation. The type of reentrancy specified the first time a handler name appears in a CONTROLLER or DEVICE statement is used for the entire system.

The CDT per unit definition table (UDT) specification applies until another HANDLER keyword or CONTROLLER statement is processed. Nonpresent devices can be configured. When the devices are added, a warm start with the devices marked online allows them to be used.

The null device, NU, must be included in every configuration.

Notes:

Extended I/O handlers default to system reentrant handlers.

GPMC device handlers default to system level reentrant handlers.

A CDT is generated each time a HANDLER keyword appears, except on the first DEVICE statement following a CONTROLLER statement with a HANDLER specification.

GPMCs support a maximum of 16 subaddresses per controller.

For XIO devices, the subaddress specified as the *aa* parameter is 2 hexadecimal digits. The first digit is the controller address and the second digit is the device address. For IOP devices, the digit specified for the controller address must be equal to the digit specified in the CONTROLLER directive SUBCH parameter. Except for floppy disks, all F-class disks must specify an even device address. The device address is determined by the unit address plug installed in the drive or by switches in the drive.

The device address is the unit address multiplied by 2 and converted to its hexadecimal equivalent (for example, unit address 1 is device address 2, unit address 7 is device address E). If more than one device is configured using one DEVICE directive, the device address and the increment (*inc*) must be even numbers. F-class cartridge module drives adhere to the same conventions. SYSGEN automatically configures the captive media portion of the drive at the next sequential odd device address. Once configured, cartridge module drives are treated as individual devices. The removable media portion is treated as a moving head disk at the even device address, and the captive media portion is treated as a fixed head disk at the next sequential odd device address.

7.6.16.1 DEVICE Syntax for Disks

Syntax

DEVICE={*aa* | (*aa*,*n*,*inc*)} [**,DEAL**]
[**,DISC=**{*mdsize* | (*mdsize* [,**D**]) | *devcode* | (*devcode* | [,**M**|**P**|**D**])}] [**,DTC=***tt*]
[**,HANDLER=**{*name* | (*name*[,**I**|,**S**] [,**C**])}] [**,IOQ=***mode*] [**,OFF**]
[**,PHYSA=***ccaa*] [**,SHR**] [**,START=***start*]

aa — is the 2-digit hexadecimal device subaddress. The first digit is the controller address; the second digit is the device address.

n — is an optional parameter specifying the decimal number of devices starting at the subaddress

inc — is an optional parameter specifying the hexadecimal address increment for each additional device. If not specified, the default is 1.

[**,DEAL**] — specifies that memory for the memory disk is not allocated at system initialization. DEAL is valid only for single-ported memory disk. If DEAL is supplied with a dual-ported memory disk, it is ignored.

[**,DISC=**{*mdsize* | (*mdsize* [,**D**]) | *devcode* | (*devcode* | [,**M**|**P**|**D**])}]

- *mdsize* — specifies the size in KBs if the device is a memory disk
- *devcode* — is the 5-character device mnemonic for disk storage devices. It is not required for devices other than disk. See Table 7-2.
- **M** — specifies the device is a multiport disk that is compatible with an MPX-32 Revision 3.3 or later system
- **P** — specifies the device is a multiport disk that is compatible with an MPX-32 Revision 3.2C or earlier system
- **D** — specifies the device is a dual-ported disk (for compatibility only)

[**,DTC=***tt*] — *tt* is a 2-character device mnemonic (see Table 7-1). If not specified, the device mnemonic specified on the associated CONTROLLER directive is used.

[,**HANDLER=**{*name* | (*name*[,I|,S] [,C])}]

name — is the 1- to 8-character handler name (see Table 7-1). If not specified, a default handler is assigned in the following manner:

- If the device has a default handler associated with it, that handler is assigned to the device.
- If the device does not have a default handler associated with it, the device is assigned the same handler as the device which was most recently defined to the subchannel to which this device is being defined (*aa* above). If no device has yet been defined to this subchannel, then the handler associated with the CONTROLLER directive for the subchannel is assigned.

An error is generated if an invalid handler is selected by default.

This parameter must be specified for a memory disk (H.MDXIO).

I — specifies interrupt priority level reentrancy, one copy per channel

S — specifies system level reentrancy, one copy per system

C — specifies one CDT for each UDT

[,**IOQ=***mode*]

is used by the IOP and GPMC to indicate the I/O queue entries are to be linked from the UDT (IOQ=DEV) or from the CDT (IOQ=CONT). The default mode is IOQ=CONT. However, most handlers provide their own initialization for this parameter and ignore any value specified by SYSGEN. Therefore, this parameter should not be used for any standard MPX-32 handler supplied by Encore unless such a requirement is specified in the installation instructions for that handler.

[,**OFF**] specifies that the device or devices described by this directive are to be SYSGENed in an offline state

[,**PHYSA=***ccaa*]

specifies the physical (bus) channel address and device subaddress for devices. Used when logical channel and subaddress do not match the physical channel and subaddress.

ccaa — is the 2-digit hexadecimal channel address and 2-digit hexadecimal device subaddress

[,**SHR**] specifies the device is a shared device

[,**START=***start*]

start specifies the decimal start map block number of the memory disk. If not specified, default is the highest possible address in memory.

Notes:

Memory disks must be configured on the even subaddresses of channel 00. The subaddress of the memory disk cannot be the same as the null device subaddress.

Table 7-2
Disk Device Codes

Disk	Disk Code
Reserved	FE004
1.2MB floppy disk - Class F device	FL001
40MB moving head disk - Class F device	MH040
80MB moving head disk - Class F device	MH080
160MB moving head disk - Class F device	MH160
300MB moving head disk - Class F device	MH300
340MB (Winchester) moving head disk - Class F device	MH340
600MB moving head disk - Class F device	MH600
5MB fixed head disk - Class F device	FH005
32MB cartridge module disk - Class F device	CD032
Any nonfloppy disk - Class F device	ANY

7.6.16.2 DEVICE Syntax for Console, Terminals and 8-Line Serial Printers

Syntax

DEVICE={*aa* | (*aa*,*n*,*inc*)} [,**CACHE**] [,**DTC=***tt*] [,**FEOP**] [,**FULL**]
[,**HANDLER=**{*name* | (*name*[,II,S] [,C])}] [,**IOQ=***mode*] [,**LINSIZ=***x*] [,**OFF**]
[,**PAGE=***y*] [,**PHYSA=***ccaa*] [,**SLPR**]

aa — is the 2-digit hexadecimal device subaddress. The first digit is the controller address; the second digit is the device address.

n — is an optional parameter specifying the decimal number of devices starting at the subaddress

inc — is an optional parameter specifying the hexadecimal address increment for each additional device. If not specified, the default is 1.

[,**CACHE**] — specifies a cache device. This option is not necessary if the associated CONTROLLER statement includes CACHE.

[,**DTC=***tt*] — *tt* is a 2-character device mnemonic (see Table 7-1). If not specified, the device mnemonic specified on the associated CONTROLLER directive is used.

[,**FEOP**] — specifies that J.SOUT inhibits the normal initial form feed and terminates all printing with an additional form feed, if specified for LP device. FEOP is applicable only on electrostatic printers and laser printers that retain the last page of output until a form feed (EJECT) is received.

[,FULL] allows full-duplex mode of operation if FULL and NOECHO are specified in LOGONFLE. This causes a UDT to be created for both the read and write subaddresses even though only the read subaddress is specified. Valid only for 8-line asynch devices.

[,HANDLER={*name*|(*name*[,I|,S] [,C])}]

name is the 1- to 8-character handler name (see Table 7-1). If not specified, a default handler is assigned in the following manner:

- If the device has a default handler associated with it, that handler is assigned to the device.
- If the device does not have a default handler associated with it, the device is assigned the same handler as the device which was most recently defined to the subchannel to which this device is being defined (*aa* above). If no device has yet been defined to this subchannel, then the handler associated with the CONTROLLER directive for the subchannel is assigned.

An error is generated if an invalid handler is selected by default.

I specifies interrupt priority level reentrancy, one copy per channel

S specifies system level reentrancy, one copy per system

C specifies one CDT for each UDT

[,IOQ=*mode*]
is used by the IOP and GPMC to indicate the I/O queue entries are to be linked from the UDT (IOQ=DEV) or from the CDT (IOQ=CONT). The default mode is IOQ=CONT. However, most handlers provide their own initialization for this parameter and ignore any value specified by SYSGEN. Therefore, this parameter should not be used for any standard MPX-32 handler supplied by Encore unless such a requirement is specified in the installation instructions for that handler.

[,LINSIZ=*x*] *x* specifies the number of characters per line for TSM devices. The default is 80.

[,OFF] specifies that the device or devices described by this directive are to be SYSGENed in an offline state

[,PAGE=*y*] *y* specifies the number of lines per screen for TSM devices or number of lines per page for listed output devices. Zero prevents `ENTER CR FOR MORE`. The default is 24 lines.

[,PHYSA=*ccaa*]
specifies the physical (bus) channel address and device subaddress for devices. Used when logical channel and subaddress do not match the physical channel and subaddress.

ccaa is the 2-digit hexadecimal channel address and 2-digit hexadecimal device subaddress

[,**SLPR**] indicates the device is a serial printer. See Notes.

Notes:

The CACHE parameter can be applied to ACM devices (H.F8XIO), to quadruple the normal I/O time out. This is intended for slow spooling devices with large internal buffers using XON/XOFF (WXON) flow control. The CACHE parameter can also be applied to a buffered tape processor (BTP), but not to the cache disk accelerator.

When SLPR is specified: the HANDLER parameter must equal H.F8XIO; the LINESIZ and PAGE parameter must be used (suggested settings LINSIZ=133 and PAGE=60); the DEVICE FULL parameter must not be specified. If the printer has a large buffer, the DEVICE CACHE parameter should be specified to extend the normal time-out. This will account for long I/O wait times (XOFF) while the buffer is purging.

7.6.16.3 DEVICE Syntax for Tape Drives

Syntax

DEVICE={*aa* | (*aa*,*n*,*inc*)} [,**ANSI**] [,**CACHE**] [,**DTC=***tt*]
[,**HANDLER=**{*name* | (*name*[,**II**,**S**] [,**C**])}] [,**IOQ=***mode*] [,**OFF**] [,**PHYSA=***ccaa*]
[,**QITD**] [,**SHR**]

aa is the 2-digit hexadecimal device subaddress. The first digit is the controller address; the second digit is the device address.

n is an optional parameter specifying the decimal number of devices starting at the subaddress

inc is an optional parameter specifying the hexadecimal address increment for each additional device. If not specified, the default is 1.

[,**ANSI**] specifies this drive is used only for ANSI tape processing. Refer to Reference Manual Volume II, Chapter 7, for details. This parameter does not restrict the system administrator.

[,**CACHE**] specifies a cache device. This option is not necessary if the associated CONTROLLER statement includes CACHE.

[,**DTC=***tt*] *tt* is a 2-character device mnemonic (see Table 7-1). If not specified, the device mnemonic specified on the associated CONTROLLER directive is used.

[,HANDLER={*name*|(*name*[,I|,S] [,C])}]

name is the 1- to 8-character handler name (see Table 7-1). If not specified, a default handler is assigned in the following manner:

- If the device has a default handler associated with it in Table 7-1, that handler is assigned to the device.
- If the device does not have a default handler associated with it, the device is assigned the same handler as the device which was most recently defined to the subchannel to which this device is being defined (*aa* above). If no device has yet been defined to this subchannel, then the handler associated with the CONTROLLER directive for the subchannel is assigned.

An error is generated if an invalid handler is selected by default.

I specifies interrupt priority level reentrancy, one copy per channel

S specifies system level reentrancy, one copy per system

C specifies one CDT for each UDT

[**,IOQ=***mode*]
is used by the IOP and GPMC to indicate the I/O queue entries are to be linked from the UDT (IOQ=DEV) or from the CDT (IOQ=CONT). The default mode is IOQ=CONT. However, most handlers provide their own initialization for this parameter and ignore any value specified by SYSGEN. Therefore, this parameter should not be used for any standard MPX-32 handler supplied by Encore unless such a requirement is specified in the installation instructions for that handler.

[**,OFF**] specifies that the device or devices described by this directive are to be SYSGENed in an offline state

[**,PHYSA=***ccaa*]
specifies the physical (bus) channel address and device subaddress for devices. Used when logical channel and subaddress do not match the physical channel and subaddress.

ccaa is the 2-digit hexadecimal channel address and 2-digit hexadecimal device subaddress

[**,QITD**] specifies a quarter-inch tape drive

[**,SHR**] specifies the device is a shared device

Notes:

The CACHE parameter can be applied to ACM devices (H.F8XIO), to quadruple the normal I/O time out. This is intended for slow spooling devices with large internal buffers using XON/XOFF (WXON) flow control. The CACHE parameter can also be applied to a buffered tape processor (BTP), but not to the cache disk accelerator.

7.6.16.4 DEVICE Syntax for IOP Printers and Null Device

Syntax

DEVICE={*aa* | (*aa*,*n*,*inc*)} [**,CACHE**] [**,DTC=***tt*] [**,FEOP**]
[**,HANDLER=**{*name* | (*name*[,II,S] [,C])}] [**,IOQ=***mode*] [**,OFF**] [**,PHYSA=***ccaa*]
[**,SHR**] [**,SPOOL=**(*code*,*code*,...)]

aa — is the 2-digit hexadecimal device subaddress. The first digit is the controller address; the second digit is the device address.

n — is an optional parameter specifying the decimal number of devices starting at the subaddress

inc — is an optional parameter specifying the hexadecimal address increment for each additional device. If not specified, the default is 1.

[**,CACHE**] specifies a cache device. This option is not necessary if the associated CONTROLLER statement includes CACHE.

[**,DTC=***tt*] *tt* is a 2-character device mnemonic (see Table 7-1). If not specified, the device mnemonic specified on the associated CONTROLLER directive is used.

[**,FEOP**] specifies that J.SOUT inhibits the normal initial form feed and terminates all printing with an additional form feed, if specified for LP device. FEOP is applicable only on electrostatic printers and laser printers that retain the last page of output until a form feed (EJECT) is received.

[**,HANDLER=**{*name* | (*name*[,II,S] [,C])}]

name — is the 1- to 8-character handler name (see Table 7-1). If not specified, a default handler is assigned in the following manner:

- If the device has a default handler associated with it in Table 7-1, that handler is assigned to the device.
- If the device does not have a default handler associated with it, the device is assigned the same handler as the device which was most recently defined to the subchannel to which this device is being defined (*aa* above). If no device has yet been defined to this subchannel, then the handler associated with the CONTROLLER directive for the subchannel is assigned.

An error is generated if an invalid handler is selected by default.

I specifies interrupt priority level reentrancy, one copy per channel

S specifies system level reentrancy, one copy per system

C specifies one CDT for each UDT

[,**IOQ=***mode*]
is used by the IOP and GPMC to indicate the I/O queue entries are to be linked from the UDT (IOQ=DEV) or from the CDT (IOQ=CONT). The default mode is IOQ=CONT. However, most handlers provide their own initialization for this parameter and ignore any value specified by SYSGEN. Therefore, this parameter should not be used for any standard MPX-32 handler supplied by Encore unless such a requirement is specified in the installation instructions for that handler.

[,**OFF**] specifies that the device or devices described by this directive are to be SYSGENed in an offline state

[,**PHYSA=***ccaa*]
specifies the physical (bus) channel address and device subaddress for devices. Used when logical channel and subaddress do not match the physical channel and subaddress.

ccaa is the 2-digit hexadecimal channel address and 2-digit hexadecimal device subaddress

[,**SHR**] specifies the device is a shared device

[,**SPOOL=**(*code,code,...*)]
indicates the device is available for automatic selection as the destination device for spooled printed and punched output. *code* consists of 2-character codes separated by commas. The codes and their use for output are:

BL	batch	SLO
BB	batch	SBO
RL	real-time	SLO
RB	real-time	SBO

Notes:

The CACHE parameter can be applied to ACM devices (H.F8XIO), to quadruple the normal I/O time out. This is intended for slow spooling devices with large internal buffers using XON/XOFF (WXON) flow control. The CACHE parameter can also be applied to a buffered tape processor (BTP), but not to the cache disk accelerator.

7.6.17 DISP Directive

The DISP directive determines the number of entries in the dispatch queue. One entry is used for each concurrently operating task. Each entry requires 64 words of resident image storage. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

DISP=*entries*

entries specifies the decimal number of entries in the dispatch queue. This value cannot exceed 255 and must be at least 8. If not specified, the default is 10.

7.6.18 DPTIMO Directive

The DPTIMO directive specifies a time-out value for resource delays encountered when attempting to access a multiprocessor, shared-volume resource. If this directive is not specified, the system assigns a sufficient number of time units to result in a delay a maximum of 1 second long. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

DPTIMO=*num*

num is the decimal number of time units to wait before timing out. Because a delay request is likely to be entered within a time unit, *num* should be a value large enough to result in delay one time unit longer than the actual maximum delay desired. For example, if NTIM is 1 and MTIM is 60, DPTIMO must be 61 (1 second plus 1 time unit) to ensure a maximum delay of 1 second. (For more information on time units, see the MTIM Directive and NTIM Directive section of this chapter.)

7.6.19 DPTRY Directive

The DPTRY directive specifies the decimal number of times that a system tries to obtain a multiprocessor resource before returning a denial. If this directive is not specified, the default is 0, which causes the system to try until the resource is obtained. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

DPTRY=*num*

num is the decimal number of tries to obtain a multiprocessor resource.

7.6.20 DTSAVE Directive

The DTSAVE directive specifies the decimal number of minutes to elapse before J.DTSAVE is resumed. If this directive is not specified, the default time is 5 minutes. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

DTSAVE=*time*

time is the number of minutes to elapse before J.DTSAVE resumes.

7.6.21 //END Directive

The //END directive is required as the last directive in the SYSGEN directives file.

Syntax

//END

7.6.22 ENDPGOUT Directive

The ENDPGOUT directive specifies the maximum percentage of total memory desired for free pages on a demand page system, the upper page-out threshold. J.SWAPR stops writing pages queued for page-out to the swap file when the ENDPGOUT value is reached or the queues are empty.

If NODEMAND is specified, this directive is ignored. If this directive is not specified and demand page processing is supported, the default is 25% of the total available memory on the system.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

ENDPGOUT=*xx*

xx is a percentage (00 through 100) of the total available memory (SRAM only) on the system. H.SINIT converts this percentage to the correct number of pages (map blocks). *xx* must be greater than the value specified for BEGPGOUT.

7.6.23 EXTDMPX Directive

The EXTDMPX directive can be used on a system wide basis to designate the starting address of the TSA and extended MPX-32 (if configured). This directive appears under the //SOFTWARE /PARAMETERS subsection.

This directive applies only to the TSA when extended MPX is not configured. Refer to Reference Manual Volume I, Chapter 3, for information on how to configure extended MPX-32.

The NOTSA or TSA option is ignored when the load module has been cataloged in the compatible mode, or using the TSA keyword in the cataloger EXTDMPX directive. The NOTSA or TSA option is effective only when the load module has been cataloged using the SYSTSA keyword in the Cataloger EXTDMPX directive. When this requirement has been met, a TSM or M.PTSK request will override the SYSGEN request.

If the EXTDMPX directive is not used, the default is MINADDR and NOTSA. This directive establishes the default TSA and extended MPX-32 logical starting address unless overridden via the CATALOG, TSM, or M.PTSK assignments.

Syntax

EXTDMPX={*logmapbl* | **MAXADDR** | **MINADDR**} **[,NOTSA | ,TSA]**

logmapbl is a decimal value between 64 and 2047 that specifies a starting map block in the task's logical address space where the TSA (optionally) and extended MPX-32 (if configured) are positioned. The NOTSA/TSA keyword controls positioning of the TSA.

MAXADDR positions the TSA (optionally) and extended MPX-32 (if configured) at the top of the the task's logical memory. The NOTSA/TSA keyword controls positioning of the TSA.

MINADDR positions the TSA and extended MPX-32 (if configured) at the bottom of the task's logical memory above MPX-32 (when mapped in), and below the task's DSECT. The TSA keyword defaults to NOTSA for MINADDR.

NOTSA directs the logical position of the TSA to be above MPX-32 (when mapped in) and below extended MPX-32 (if configured and at MINADDR), and below the task's DSECT.

TSA directs the repositioning of the TSA in accordance with the MAXADDR, or *logmapbl* specification used. For MAXADDR the TSA is located at the top of the task's logical memory, followed by extended MPX-32 (if configured). For *logmapbl*, the TSA logically starts at *logmapbl*, followed by extended MPX-32 (if configured).

Note: An error, `***INVALID KEYWORD`, is displayed on the user's terminal (or printed to the listed output) if `NOTSA` or `TSA` keywords are incorrectly spelled. SYSGEN aborts and no image is built.

At runtime, values for `MAXADDR` or *logmapbl* that conflict with the task's code, data, or partition memory requirements cause an abnormal termination in the task activation.

7.6.24 /FILES Directive

The /FILES directive designates the system files subsection of the //SOFTWARE section. This section is included for compatibility only.

Syntax

/FILES

7.6.25 FLTSIZE Directive

The FLTSIZE directive is included for compatibility and is ignored by SYSGEN. Items following this directive on the same line are ignored.

Syntax

FLTSIZE

7.6.26 //HARDWARE Directive

The //HARDWARE directive indicates the beginning of the hardware configuration section of directives. This directive is required.

Syntax

//HARDWARE

7.6.27 HELP Directive

The HELP directive specifies the volume and/or directory containing help files. The default is @SYSTEM ^(HELP) directory. The directive also activates J.HLP. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

HELP [,**VOL=***volname*] [,**DIR=***dirname*]

volname is the volume where help files are located. If not specified, the default is the SYSTEM volume.

dirname is the directory where help files are located. If not specified, the default is the HELP directory.

7.6.28 /INTERRUPTS Directive

The /INTERRUPTS directive designates the interrupt subsection of the //HARDWARE section. This directive is required.

Syntax

/INTERRUPTS

7.6.29 IOQPOOL Directive

The IOQPOOL directive specifies the number of words to reserve at SYSGEN for the IOQ memory pool. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is specified, the IOQ memory pool is used only for IOQ requests. IOQ memory pool is required for each preallocated IOQ and for each I/O in progress.

To estimate the minimum size to request for the IOQ memory pool, calculate the average IOQ size (26 words plus the length in words of the associated I/O command list (IOCL)). (The length of the IOCL varies according to the I/O operation and the device.) The calculated size is rounded up to a doubleword. Then multiply the average IOQ size by the estimated number of IOQs in use at any one time.

The size of the IOQPOOL should be greater than the worst case I/O load. If the IOQPOOL becomes used up, causing rollover to miscellaneous memory pool, the probability of deadlock occurring from requesting memory pool increases. When rollover to miscellaneous memory pool occurs, a bit (C.ROLIOQ) is set in the communications region.

If the MNWI directive is not specified, unprivileged tasks can have one wait I/O or five nowait I/Os outstanding at a given time. Privileged tasks can have any number of nowait I/Os outstanding.

Syntax

IOQPOOL=*n*[**,PERMIOQ** [**,NOROLL**]]

n is the decimal number of words to reserve. If *n* is not specified, an IOQ memory pool is not reserved and the memory pool that is specified by the POOL directive handles any IOQ requests.

PERMIOQ specifies static IOQ. This parameter is valid only when parameter *n* is greater than 0. PERMIOQ enables an IOQ to be allocated when the file is opened and deallocates that IOQ when the file is closed. If PERMIOQ is not specified, IOQs are allocated as described in the Operating System Memory Allocation section, in Chapter 3 of MPX-32 Reference Manual, Volume I.

NOROLL specifies no rollover to the memory pool specified by the POOL directive for static IOQs when the IOQ memory pool is full.

7.6.30 IPU Directive

The IPU directive specifies an IPU will be configured into the system. This directive appears under the //HARDWARE /PARAMETERS subsection.

Syntax

IPU

7.6.31 ITIM Directive

The ITIM directive provides the expiration time interval of the RTOM interval timer. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is not specified, the default time interval is 38.4 microseconds.

Syntax

ITIM=*value*

value is the quantity expressed in tenths of microseconds of an RTOM interval-timer time quantum. For example, 38.4 microseconds is represented as 384.

7.6.32 ITLB Directive

The ITLB directive generates an Indirectly Connected Task Linkage Block. This directive appears under the //SOFTWARE /PARAMETERS subsection. One ITLB directive is required for each indirectly linked task concurrently active in the system.

Syntax

ITLB=*intlevel*

intlevel is the 2-character hexadecimal interrupt priority level where the task will be indirectly connected

7.6.33 JOBS Directive

The JOBS directive specifies the maximum number of batch jobs that can be concurrently active. Any number is valid. However, a large number can cause an increase in system response time due to increased system overhead associated with J.TSM processing. Therefore, 10 is suggested. If this directive is not specified, the default is 1.

This directive appears under the //SOFTWARE /TABLES subsection.

Syntax

JOBS=*number*

number is the number of entries in the job table.

7.6.34 KTIMO Directive

The KTIMO directive specifies the number of seconds the kill directive will attempt to abort a task before it kills it. If 0 is specified, an immediate kill is performed. If this directive is not specified, a kill is performed after 10 seconds. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

KTIMO=*number*

number is the decimal number of seconds to attempt an abort

7.6.35 LOD Directive

The LOD directive specifies a system listed output device that is the default device for SLO. This directive is required if a line printer is configured. This directive appears under the //HARDWARE /SYSDEVS subsection.

Syntax

LOD=*devmnc* [,**IBP**]

devmnc is the 2-character device mnemonic (see Table 7-1), the 2-digit hexadecimal channel number, and the 2-digit hexadecimal device subaddress.

IBP inhibits banner page produced on the SLO device

7.6.36 LOGON Directive

The LOGON directive specifies whether users may log into MPX-32 multiple times using the same owner name. It also provides an option for restricting the owner name "SYSTEM" from logging on to more than one TSM device at a time. If the LOGON directive is not specified, J.TSM allows one logon per owner name.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

LOGON={MULTI [,NOSYS] | SINGLE }

MULTI J.TSM allows a user to log on multiple times using the same owner name

[,NOSYS] restricts the owner name "SYSTEM" to one logon at a time

SINGLE J.TSM allows one logon per owner name

When an owner name is logged on multiple times, all messages and signals directed to that owner name go to all terminals logged on with that owner name. (See the BATCHMSG directive in this chapter for information on inhibiting task messages.)

Note: This directive was previously called LOGIN. "LOGIN" in place of "LOGON" remains valid syntax.

7.6.37 MACHINE Directive

The MACHINE directive indicates the type of computer for which the resultant system is being configured. If this directive is not specified, any CONCEPT/32 machine can be used.

If the type of computer specified is a CONCEPT 32/2000, the system file OH.32_OUT is dynamically assigned as SYSGEN's object module file (OBJ) and a mapped out image is built. If any other type of computer is specified or if the entire directive is omitted, OH.32 is assigned to OBJ and a mapped in image is built. These file assignments can be overridden by assigning OBJ before running SYSGEN. (See section 7.3 of this chapter for more information on assigning the object module file.) If this is done, the MACHINE directive indicates only the computer type; the image is built from the specified object module file. Any conflicts are resolved by SYSGEN in a later phase or by H.SINIT.

The MACHINE directive controls the automatic console configuration functionality. Automatic console configuration is the process of dynamically configuring the operator console for a CONCEPT 32/2000 system. This console will be dynamically configured at the CPU's F-class channel.

If no MACHINE directive is specified or if a CONCEPT 32/2000 is specified (`MACHINE=2000`), the operator console will be dynamically configured when the resultant system image is booted on a CONCEPT 32/2000 machine. The operator console will not be dynamically configured if a computer type other than a CONCEPT 32/2000 is specified or the system image is booted on a machine other than a CONCEPT 32/2000.

If this directive is specified, it must be the first directive in the //HARDWARE /PARAMETERS subsection.

Syntax

MACHINE=*type*

type is the machine type 3267, 3287, 3297 or 2000.

7.6.38 MAPOUT Directive

The MAPOUT directive designates that the default execution mode for all tasks cataloged with the ENVIRONMENT keyword SYSMAP is executed with MPX-32 mapped out of the task's address space. If neither the MAPOUT or NOMAPOUT directive is specified the compatible or NOMAPOUT mode is the default.

If the MAPOUT directive is present with a MACHINE directive specifying any type other than 2000, or if both MAPOUT and NOMAPOUT are specified, a `***CONFLICTING OPTIONS` error is generated by SYSGEN. If this error occurs, SYSGEN aborts and no image is produced.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

MAPOUT

7.6.39 MDT Directive

The MDT directive enables rapid file allocation through a memory resident descriptor table (MDT). The MDT's parameters are defined by this directive, and module H.MDT is included in the resident operating system. This directive appears under the //SOFTWARE /TABLES subsection.

Syntax

MDT=*n* [,**BLOC=***b*]

n specifies the decimal number of MDT entries. MDT entries are 192 words in length. To accommodate collision resolution, the actual number allocated is 25% greater than the specified number.

[,**BLOC=***b*] specifies the decimal starting physical map block number. If not specified, the default is the highest contiguous memory available.

If the specified starting map block is not available, initialization processing halts and an abort code is displayed at system initialization.

7.6.40 /MEMORY Directive

The /MEMORY directive designates the memory subsection of the //HARDWARE section. This directive is required.

Syntax

/MEMORY

7.6.41 MMSG Directive

The MMSG directive specifies the maximum number of no-wait messages that an unprivileged task can send. If this directive is not specified, the default is 5. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

MMSG=*num*

num is the maximum number of no-wait messages to be sent by a unprivileged task.

7.6.42 MNWI Directive

The MNWI directive specifies the maximum number of no-wait I/O requests that can be concurrently outstanding for a task. If this directive is not specified, the default is 5. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

MNWI=*num*

num is the maximum number of no-wait I/O requests that can be concurrently outstanding for a task.

7.6.43 MODE Directive

The MODE directive requests the following special system operations:

- Continuous Batch — Batch stream input from SID is processed until the job control statement $$$ is encountered. All $$ job control statements are ignored.
- Inhibit Banner Page — Suppresses the banner page produced by system output tasks when processing SLO files.
- Inhibit Mount Message — Suppresses the mount message produced by J.MOUNT. This specification is not valid with multivolume magnetic tape operations.
- Dump — Performs a dump if an independent task aborts.
- Scratchpad Locations — Does not zero unused CPU scratchpad locations at IPL.
- Inhibit Operator Intervention — Suppresses all prompts normally displayed on the system console during system initialization.
- Real-time Accounting — Turns real-time accounting on or off.

Specify a separate MODE directive for each mode desired. If no MODE directive is specified, only real-time accounting is functional in the resulting system image. If this directive is specified, it appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

MODE=*mode*

mode	is any of the following:	
	DUMP	sets dump request for aborting real-time tasks
	LSPA	inhibits zeroing of unused scratchpad locations during IPL processing. If not specified, the IPL process zeroes all unused CPU scratchpad locations.
	ONRA	turns real-time accounting on. (This is the default.)
	OFRA	turns real-time accounting off
	SCBT	sets continuous batch
	SIBP	sets inhibit banner page
	SIMM	sets inhibit mount messages for nonmultivolume magnetic tape operations
	SNOP	sets inhibit operator intervention

7.6.44 MODULE Directive

The MODULE directive names optional user modules to be included in the MPX-32 resident image. One MODULE directive is required for each user module to be included. This directive appears under the //SOFTWARE /MODULES subsection.

Syntax

MODULE=(*name,module,entpoints***)**

name	is the 1- to 8-character ASCII module name. The name cannot contain a comma, an equal sign, or a left or right parenthesis. The module name N.ACXRF is reserved for use only as module number 12.
module	is a 2-digit decimal number representing the internal identification number of the module. Modules 00 through 12 are reserved for MPX-32 modules. A module number of 12 is allowed only if the module name is N.ACXRF.
entpoints	is the hexadecimal number of entry points contained in this module. The last entry point of each user-supplied module is an initialization entry point called by SYSGEN during construction of the MPX-32 image and is overlaid following execution. This entry point should not be included in *entpoints*.

Module initialization must include the system macros M.EIR, M.XIR, M.MODT, and M.SVCT.

7.6.45 /MODULES Directive

The /MODULES directive designates the modules subsection of the //SOFTWARE section. If the MODULE directive is not specified, this directive is not required.

Syntax

/MODULES

7.6.46 MRUN Directive

The MRUN directive specifies the maximum number of no-wait run requests to be sent by a task. If this directive is not specified, the default is 5. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

MRUN=*num*

num is the maximum number of run requests that can be sent by a task.

7.6.47 MSGPOOL Directive

The MSGPOOL directive specifies the number of words to reserve at SYSGEN for the MSG memory pool. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is specified, any message or run request queue (MRRQ) requests use this memory pool. Each MRRQ request corresponds to a message or run request sent by a task. To estimate the minimum size to request for MSG memory pool, calculate the average message send size or message return size (whichever is greater) and add 8 words for a message header. Multiply this average size by the estimated number of messages at any instant. The specified size is rounded up to a doubleword. Messages are sent from:

- J.TSM or OPCOM to physically mount volumes
- J.TSM to J.SSIN and back to batch tasks
- J.TSM to J.SOUT and J.SOEX to process spooled output

When rollover to miscellaneous memory pool occurs, a bit (C.ROLMSG) is set in the communications region.

Syntax

MSGPOOL=*n***[,NOROLL]**

n is the decimal number of words to reserve. If *n* is not specified, the MSG memory pool is not reserved and the memory pool that is specified by the POOL directive handles all MRRQ requests.

[,**NOROLL**] specifies no rollover to the memory pool specified by the POOL directive for MRRQ requests when the MSGPOOL is full

7.6.48 MTIM Directive

The MTIM directive provides the number of real-time clock interrupts per second. If this directive is not specified, the default is 60. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

MTIM=*number*

number is the number of real-time clock interrupts per second.

Note: The number of time units per second is the value of NTIM divided by the value of MTIM. See the NTIM directive in this chapter for more information.

7.6.49 NAME Directive

The NAME directive defines static Datapool or Global Common memory partitions. This directive appears under the //SOFTWARE /PARTITION subsection.

If multiple static partitions are defined within a map block, only one partition can be included in the task's logical address space at a given time. One NAME directive is required for each memory partition desired.

Syntax

NAME=*name***,SIZE=***np***,STRTPG=***sp***,MAP=***pm*

name is the 1- to 8-character partition name

np is the decimal number of 512-word protection granules for the partition

sp is the logical hexadecimal starting protection granule for the partition

pm is the starting physical decimal map block number. A map block is 2KW.

7.6.50 NOANSI Directive

The NOANSI directive excludes support for ANSI labeled tapes from the system image. This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

NOANSI

7.6.51 NOBASE Directive

The NOBASE directive excludes support in H.EXEC, H.IP00, H.IP04, H.IP0F, and H.TAMM for the execution of tasks utilizing base mode addressing from the system image. This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

NOBASE

7.6.52 NOCMS Directive

The NOCMS directive automatically excludes support for MPX-1.*x* compatibility mode services (modules H.ALOC, H.FISE, H.MONS, and H.CALM) from the system image. Software that calls a removed compatibility mode system module is aborted with an SV09 abort code. Do not include this directive in the SYSGEN file if you plan to use compatiblility mode services.

This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

NOCMS

7.6.53 NODEMAND Directive

The NODEMAND directive indicates demand page processing is not allowed.

Demand page processing is supported when the mapped out object file is assigned at SYSGEN execution. If other than the mapped out object file is assigned, demand page is not supported, and this directive is ignored.

If specified, this directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

NODEMAND

7.6.54 NOLACC Directive

The NOLACC directive overrides the MPX-32 default option of tracking last date, time, and owner name of access to task level resources. This directive, if specified, appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

NOLACC

7.6.55 NOMAPOUT Directive

The NOMAPOUT directive designates that the default execution mode for all tasks cataloged with the ENVIRONMENT keyword SYSMAP is executed with MPX-32 mapped into the task's address space. If neither the MAPOUT nor NOMAPOUT directive is specified the compatible or NOMAPOUT mode is the default.

If both MAPOUT and NOMAPOUT are spectified, a `***CONFLICTING OPTIONS` error is generated by SYSGEN. If this error occurs, SYSGEN aborts and no image is produced.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

NOMAPOUT

7.6.56 NOSYSVOL Directive

The NOSYSVOL directive prevents mounting and use of the system volume. Use this directive only when the system is to be configured with no system volume. It is for use with 3.0 or later revisions of Reflective Memory System Software (RMSS). This directive forces NOCMS.

This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

NOSYSVOL

7.6.57 NOTDEF Directive

The NOTDEF directive excludes support for the TERMDEF facility from the system image. This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

NOTDEF

7.6.58 NOTSMEXIT Directive

The NOTSMEXIT directive keeps TSM active even though it is not in use. This is the default directive. If specified, it appears under the //SOFTWARE /PARAMETERS subsection. To enable exit of TSM when inactive, see the TSMEXIT Directive.

Syntax

NOTSMEXIT

7.6.59 NTIM Directive

The NTIM directive provides the number of real-time clock interrupts per time unit. If this directive is not specified, the default is 60. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

NTIM=*number*

number is the number of clock interrupts per time unit.

Note: The number of time units per second is the value of NTIM divided by the value of MTIM. See the MTIM directive in this chapter for more information.

7.6.60 OTHERS Directive

The OTHERS directive specifies access rights allowed to users who are not the owner or members of the project group associated with a partition previously defined with a NAME directive. If more than one NAME directive has been specified, the OTHERS directive only applies to the last one specified. If this directive is not specified, default is OTHERS=(R).

If both read and write access are desired, a comma must be used as a delimiter, for example, OTHERS=(R,W).

The OTHERS directive must always be specified last when OWNER and/or PROJECT directives are also specified. The order is OWNER, PROJECT, OTHERS.

This directive appears under the //SOFTWARE /PARTITION subsection.

Syntax

OTHERS=([R] [,W] [,N])

R	specifies read access is allowed
W	specifies write access is allowed
N	specifies no access is allowed

7.6.61 /OVERRIDE Directive

The /OVERRIDE directive designates the override subsection of the //SOFTWARE section. This directive is required only if one or more directives belonging under the /OVERRIDE subsection is specified.

Syntax

/OVERRIDE

7.6.62 OWNER Directive

The OWNER directive specifies the owner name and associated access rights that apply to the partition previously defined in a NAME directive. If more than one NAME directive has been specified, the OWNER directive only applies to the last one specified. If this directive is not specified, default is OWNER=(SYSTEM,R,W).

This directive appears under the //SOFTWARE /PARTITION subsection.

Syntax

OWNER=(*name* [,R] [,W] [,N])

name	is the 1- to 8-character owner name to be associated with the partition
R	specifies read access is allowed
W	specifies write access is allowed
N	specifies no access is allowed

7.6.63 OWNERNAME Directive

The OWNERNAME directive prevents the characters of the user's owner name from echoing to the terminal screen as the user logs on. If OWNERNAME=NOECHO is specified, no terminals will echo the owner name. If OWNERNAME=ECHO is specified or if the OWNERNAME directive is not included in the SYSGEN file, all terminals will echo the owner name.

This directive appears under the //SOFTWARE /SECURITY subsection.

Syntax

OWNERNAME={NOECHO I ECHO}

NOECHO J.TSM inhibits echoing of owner names on all terminals including the system console

ECHO J.TSM allows echoing of owner names on all terminals including the system console. This is the default.

7.6.64 /PARAMETERS Directive

The /PARAMETERS directive designates the beginning of the parameters subsection of the //HARDWARE and //SOFTWARE sections. This directive is required and it must be the first subsection of the //HARDWARE and //SOFTWARE sections.

Syntax

/PARAMETERS

7.6.65 /PARTITION Directive

The /PARTITION directive designates the memory partition and/or global common subsection of the //SOFTWARE section. This directive is required only if one or more partition directives is specified.

Syntax

/PARTITION

7.6.66 PASSWORD Directive

The PASSWORD directive specifies that passwords are required to logon to MPX-32. This directive appears under the //SOFTWARE /SECURITY subsection.

Syntax

PASSWORD

7.6.67 PATCH Directive

The PATCH directive specifies a system patch area to append to the MPX-32 resident image. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is not specified, a system patch area is not reserved.

Syntax

PATCH=*number*

number specifies the hexadecimal number of bytes to be added to the resident MPX-32 system as a system patch area. If the extended mode was activated at SYSGEN this specifies the hexadecimal number of bytes to be added in the first 16KW of memory.

Note: The system patch area is restricted by SYSGEN to the first 16KW of memory. SYSGEN also restricts the area from exceeding the first 16KW of memory.

7.6.68 PCHFILE Directive

The PCHFILE directive names the permanent file which the generated system uses as its patch file. SYSGEN does not create the file; it must be supplied if patches are to be generated. If this directive is not specified, the default is M.PATCH.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

PCHFILE=*filename*

filename is the 1- to 8-character ASCII file name of the file to contain patches for the generated system.

7.6.69 POD Directive

The POD directive specifies the system punched output (SPO) device to use as the default device for SBO. This directive appears under the //HARDWARE /SYSDEVS subsection.

Syntax

POD=*devmnc*

devmnc is the 2-character device mnemonic (see Table 7-1), the 2-digit hexadecimal channel number, and the 2-digit hexadecimal device subaddress.

7.6.70 POOL Directive

The POOL directive specifies the size in words of the memory pool to be reserved at SYSGEN time. The specified size of the memory pool is rounded up to the end of the current map block. If this directive is not specified, then 1000 words are reserved for the memory pool.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

If the IOQPOOL and MSGPOOL directives are not specified, the memory pool requested by the POOL directive is the only memory pool. This memory pool is then used for all requests.

If the IOQPOOL and/or MSGPOOL directive is specified, the total memory pool size is the memory pool plus the IOQ memory pool and/or the MSG memory pool. This total is then rounded up one map block boundary.

The memory pool requested by the POOL directive is then used for miscellaneous requests (requests other than IOQ or MRRQ requests).

This memory pool is also used when there is no space available in the IOQ memory pool or the MSG memory pool, provided NOROLL was not specified for the IOQ or MSG memory pool.

Syntax

POOL=*words*

words is the decimal number of words to be reserved.

7.6.71 PRIORITY Directive

The PRIORITY directive specifies the interrupt configuration and interrupt processors to use for the target system. The interrupt processors provided are: Attention (H.IP13), Real-time Clock (H.IPCL), and CPU Interval Timer (H.IPIT). The CPU Interval Timer is directly connected to the lowest interrupt priority level in the system. If a machine directive is not specified, the lowest interrupt level is 6F.

This directive appears under the //HARDWARE /INTERRUPTS subsection.

If an IPU is SYSGENed into the system, an IPU accounting processor (H.IPUIT) is available. This processor is directly connected to the IPU accounting interval timer at any priority. The lowest available priority is recommended.

I/O channel interrupt levels on a CONCEPT/32 are 04 to 13.

Syntax

PRIORITY=*intlev*, **RTOM=(***channel,subaddress***)** [**,PROGRAM=***name*] [**,INTV**]

intlev is the 2-digit hexadecimal interrupt level. The lowest level is 6F.

channel is the 2-digit hexadecimal RTOM board or IOP RTOM function channel address

subaddress is the 2-digit hexadecimal RTOM board or IOP RTOM function subaddress, which is the one's complement of the RTOM relative physical priority for the interrupt level.

[,**PROGRAM**=*name*]
is the 1- to 8-character name of the program located in the file assigned to LFC OBJ. If the program is to be directly connected to this interrupt level and SYSGENed with the resident operating system, the name must be supplied. If tasks are to be indirectly connected to the interrupt level, no program name should be supplied. An indirectly connected task linkage block (ITLB) is defined using the ITLB directive.

[,**INTV**] indicates the level is an RTOM interval timer. A device entry, whose address is equal to the interrupt priority level, is being built in the scratchpad.

7.6.72 PROGRAM Directive

The PROGRAM directive specifies the program names of the trap processors to be configured on the resident system. This directive appears under the //HARDWARE /TRAPS subsection.

If this directive is specified, no defaults are in effect and all trap handlers to be included on the system must be listed. If it is not specified, the following defaults are used:

H.IP00	H.IP09
H.IPAS	H.IP0C
H.IP02	H.IP0F
H.IP03	H.IP10
H.IP04	H.IP13
H.IP05	H.IPHT
H.IP06	H.IPU
H.IP07	H.IPUAS (if an IPU is configured)
H.IP08	H.CPU

If the defaults are required along with other optional system modules, the USERPROG directive should be used instead of the PROGRAM directive.

Syntax

PROGRAM= (*name1, . . . name7*)

name1...name7
are 1- to 8-character ASCII program names of the trap handlers assigned to LFC OBJ, separated by commas. A maximum of 7 names can be entered per directive.

Note: Do not include H.CPU2 or H.EXEC2 in this directive, because the DELTA directive automatically selects them.

7.6.73 PROJECT Directive

The PROJECT directive specifies the project group name and associated access rights which apply to the partition previously defined with a NAME directive. If more than one NAME directive has been specified, the PROJECT directive only applies to the last one specified. This directive is optional. If not specified, the default is PROJECT=(SYSTEM,R,W).

The OWNER directive must precede PROJECT and OTHERS directives if specified. The order is OWNER, PROJECT, OTHERS.

This directive appears under the //SOFTWARE /PARTITION subsection.

Syntax

PROJECT=(*name* **[,R] [,W] [,N])**

name is the 1- to 8-character project group name associated with the partition

R specifies read access is allowed

W specifies write access is allowed

N specifies no access is allowed

7.6.74 RLWU Directive

The RLWU directive specifies whether read/lock write/unlock applies to all memory, or only uncached ranges. This directive should be specified only once. Additional RLWU directives are ignored. If this directive is not specified, read/lock write/unlock applies only to uncached ranges. This directive appears under the //HARDWARE /MEMORY subsection.

Syntax

RLWU={Y|N}

Y specifies that read/lock write/unlock (RLWU) is employed over all memory when memory has the read/lock attribute

N specifies that RLWU is not employed over all memory. RLWU still applies to uncached ranges.

7.6.75 /RMSTABLS Directive

The /RMSTABLS directive designates the Resource Management System Tables subsection of the //SOFTWARE section. This directive is required if the ARTSIZE directive is specified.

Syntax

/RMSTABLS

7.6.76 RMTSIZE Directive

The RMTSIZE directive increases the number of bytes per entry in the Resourcemark Table. If this directive is not specified, the default is 0. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

RMTSIZE=*num*

num is a decimal number between 64 and 1000.

7.6.77 SAPASSWD Directive

The SAPASSWD directive disables the PASSWORD task so that only the system administrator can change a password. This directive appears under the //SOFTWARE /SECURITY subsection.

Syntax

SAPASSWD

7.6.78 /SECURITY Directive

The /SECURITY directive designates the security subsection of the //SOFTWARE section. This directive is required only if one or more security directives is specified.

Syntax

/SECURITY

7.6.79 SEQUENCE Directive

The SEQUENCE directive names load modules to be activated in a sequential manner by SYSINIT immediately after the target system is booted. Tasks activated with this directive must complete and exit before the next task is activated.

The system console cannot be used as a TSM terminal until all specified tasks are complete.

SYSINIT checks the completion status of each task as it exits and displays a message on the system console if abnormal status was returned. A maximum of 70 bytes of call back information (in ASCII) can be sent using the receiver exit block (RXB) when exiting a run receiver task. This information, if present, is displayed by SYSINIT on the system console. The user status byte, if not zero, is also displayed in decimal on the system console.

If control switch 5 is set before booting the system, task names listed in the SEQUENCE directive will not be activated as part of the target system boot (load) process.

This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

SEQUENCE=(*name1,...name7*)

name1,...name7
: are the 1- to 8-character ASCII load module names to be activated, separated by commas. A maximum of 7 names can be entered per directive.

7.6.80 SGOSIZE Directive

The SGOSIZE directive is included for compatibility only. This directive is accepted by SYSGEN, but the results are not used in job processing. This directive appears under the //SOFTWARE /FILES subsection.

Syntax

SGOSIZE=*blocks*

blocks
: is the number of 192-word blocks of disk space to be allocated for each SGO file

7.6.81 SHARE Directive

The SHARE directive specifies the number of entries in the shared memory table (SMT). Each entry defines a shared memory area; for example, CSECT, Global Common, Datapool, or Shared Image. A minimum of 2 entries must be specified to use the Text Editor and Volume Manager. If this directive is not specified, the default for non-extended and mapped out images is one SMT entry for the .MPXTBLS partition. The default for an extended MPX-32 images is two SMT entries, for the .extdmpx and .MPXTBLS partitions.

SYSGEN automatically increments C.SMTN for these partitions.

This directive appears under the //SOFTWARE /TABLES subsection.

Syntax

SHARE=*number*

number is the number of entries in the shared memory table. It must be sufficient to define all static and dynamic partitions.

7.6.82 SID Directive

The SID directive specifies a system input device (SID) to use as the default device in related OPCOM commands. This directive appears under the //HARDWARE /SYSDEVS subsection.

Syntax

SID=*devmnc* [**,DENSITY=***density*] [**,PARITY=***parity*]

devmnc is the 2-character device mnemonic (see Table 7-1), the 2-digit hexadecimal channel number, and the 2-digit hexadecimal device subaddress.

[**,DENSITY=***density*]
is used only for 7-track magnetic tape. H is for high density. L is for low density.

[**,PARITY=***parity*]
is used only for 7-track magnetic tape. E for even parity. O for odd parity.

7.6.83 SIZE Directive

The SIZE directive specifies the memory configuration of the target system. This directive is required. The SIZE directive appears under the //HARDWARE /MEMORY subsection.

Using multiple SIZE directives, memory types must be specified in order of configuration in memory; that is, from low address to high address.

The SIZE directive now has a second syntax which is preferred to the original syntax. This new syntax allows the user to mix $(MS)^2$ and RMS shared memory and to restrict the allocation of multiprocessor shared memory to only those tasks that physically request it. The original SIZE directive syntax is still accepted.

7.6.83.1 Preferred SIZE Directive Syntax

Syntax

SIZE=*blocks* **, TYPE=***mem* [**,RESERVED**] [**, SHARED=**{**N|Y** [**, INIT=** {**Y|N**}] [**,RMS=***addr*]}] [**,CACHE=**{**Y|N**}]

SIZE=*blocks*
: *blocks* is the memory size in decimal number of map blocks. Maximum value is 2048.

TYPE=*mem*
: *mem* is a memory type from the list below. E, H, and S are memory types that a nonbase task can request with the Catalog ENVIRONMENT directive.

Type	Description
D	SelBUS (DRAM) memory (CONCEPT 32/2000 only. See note 4.)
E	task delays until E class is available
H	task delays until H or E class is available
S	task delays until S, H, or E class is available
N	indicates absent memory
H1	is CPU shadow memory
H2	is IPU shadow memory
H3	is CPU and IPU shadow memory

[**,RESERVED**]
: reserves shadow memory for requesting tasks. This option is valid only with types H1, H2, and H3.

[**,SHARED=**{**N|Y**}]
: N specifies that multiprocessor memory is allocated to any task. Y specifies it is allocated only to tasks requesting physical memory for shared images or static partitions. N is the default.

[**,INIT=**{**Y|N**}]
: N specifies that the memory is not initialized at system startup. Y specifies to initialize shared memory at system startup. Y is the default.

[**,RMS=***addr*]
: *addr* is the hexadecimal address of the Reflective Memory System (RMS) control registers

[**,CACHE=**{**Y|N**}]
: N specifies that memory is uncached. Y specifies that memory is cached. Y is the default.

This is a multiprocessor shared memory example for two systems with 2MB of multi-processor shared memory and 4MB of non-shared memory. The two systems are to share 128KW of memory. The balance of the 2MB is divided for use by the two systems.

System 1:

```
/MEMORY
SIZE=512,TYPE=S                                4MB OF NON-SHARED
                                               MEMORY
SIZE=64,TYPE=S,SHARED=Y,INIT=N,CACHE=N         128KW TO BE SHARED
SIZE=112,TYPE=S,SHARED=Y,INIT=N,CACHE=N        224KW TO BE USED BY
                                               SYSTEM 1
SIZE=80,TYPE=N,SHARED=Y,INIT=N,CACHE=N         160KW TO BE USED BY
                                               SYSTEM 2
.
.
/PARTITION
NAME=GLOBAL01,SIZE=256,STRTPG=100,MAP=512
.
.
```

System 2:

```
/MEMORY
SIZE=512,TYPE=S
                                               MEMORY
SIZE=64,TYPE=S,SHARED=Y,INIT=N,CACHE=N         128KW TO BE SHARED
SIZE=112,TYPE=N,SHARED=Y,INIT=N,CACHE=N        224KW TO BE USED BY
                                               SYSTEM 1
SIZE=80,TYPE=S,SHARED=Y,INIT=N,CACHE=N         160KW TO BE USED BY
                                               SYSTEM 2
.
.
/PARTITION
NAME=GLOBAL01,SIZE=256,STRTPG=100,MAP=512
.
.
```

Notes:

1. SYSGEN memory directives must reflect the hardware cache configuration of the target machine.
   - On a 32/87 or 32/97 processor, the SYSGEN directives for uncaching should match the memory configuration of CPU board C.
   - On a 32/67 or CMOS processor, the hardware cache configuration is software controllable with the RLWU directive and the SIZE directive CACHE option if the memory has the proper attributes. If memory is not cache coherent, CACHE=N must be specified.
2. The CACHE=N option uncaches a physically contiguous range of memory on half-megabyte boundaries in half-megabyte increments. This range is bounded by the highest and lowest map blocks of contiguous ranges resulting from multiple SIZE directives. The lower bound is set at the first specification of the CACHE=N option and the higher bound is set at the last CACHE=N. All memory between these bounds is uncached and the range is ended by the first CACHE=Y. After this, no other CACHE=N can be specified. The CACHE option must reflect the physical attributes of the configured memory. Memory used as non-cache coherent must be specified as CACHE=N.
3. The SHARED option should be used when $(MS)^2$ or reflective memory is shared among processors. When $(MS)^2$ memory is shared, ranges within physically shared memory can be treated independently. For example, a system can be SYSGENed with range A shared by several processors and range B treated as nonshared. To do this, range A is specified as SHARED=Y on all processors and should be initialized at system startup by only one processor. Range B is specified as SHARED=N on the processor using it and as nonpresent (TYPE=N) on the other processors. Range B should be initialized at system startup by the processor using it.
4. Memory types E, H, and S must reside in SRAM memory. Type D memory must reside in DRAM. Type D memory is only valid for the CONCEPT 32/2000 processors. If DRAM is declared on any processor other than the CONCEPT 32/2000, it remains defined as type D memory. The restrictions for DRAM on a CONCEPT 32/2000 apply to DRAM on other processors.

   Once type D memory is defined, no more definitions of E, H, or S are valid. If E, H, or S memory follows the definition of D memory, the following error message is generated and no image is produced:

   ```
   ***TYPE=E, H OR S IS NOT VALID AFTER TYPE=D
   ```
5. H1, H2, and H3 are not valid for the CONCEPT 32/2000. If defined on a 32/2000, shadow memory defaults to type H.

7.6.83.2 Original SIZE Directive Syntax

Syntax

SIZE=*nn***,TYPE=***c* **,CLASS=***x* **[[,RESERVED] [,MULTI] [,MS2]]**

nn — is the size in decimal number of map blocks for phyical memory or the number of pages for memory partitions. Maximum value is 2048.

c — is memory type D, E, H, S, N, H1, H2, or H3. E, H, and S are memory types as described in Volume I. Absent memory is indicated by N. H1, H2, and H3 are shadow memory types. D is DRAM memory (CONCEPT 32/2000 only).

x — is memory class C or S:

- **C** is core memory
- **S** is semiconductor memory

[,RESERVED]
reserves shadow memory for requesting tasks. Valid only with types H1, H2, or H3.

[,MULTI] specifies the multiprocessor memory flag is set indicating this memory is not to be initialized during the startup procedure. This memory is to be uncached and have read/lock, write/unlock employed in the uncached range. It will not be initialized.

[,MS2] identifies the shared memory configured as Multiprocessor Shared Memory System $(MS)^2$ rather than the MBC/MBA Multiport Memory System. Multiprocessor Shared Memory System $(MS)^2$ is required for any dual-ported memory disk. This memory is to be cached and have read/lock, write/unlock employed. It will not be initialized. If MS2 is used, the MULTI option must also be specified.

This is a multiprocessor shared memory example for two systems with 2MB of multi-processor shared memory and 4MB of non-shared memory. The two systems are to share 128KW of memory. The balance of the 2MB is divided for use by the two systems.

System 1:

```
/MEMORY
SIZE=512,TYPE=S,CLASS=S                      4MB OF NON-SHARED
                                             MEMORY
SIZE=64,TYPE=S,CLASS=S,MULTI                 128KW TO BE SHARED
SIZE=112,TYPE=S,CLASS=S,MULTI                224KW TO BE USED BY
                                             SYSTEM 1
SIZE=80,TYPE=N,CLASS=S,MULTI                 160KW TO BE USED BY
                                             SYSTEM 2
.
.
/PARTITION
NAME=GLOBAL01,SIZE=256,STRTPG=100,MAP=512
.
.
```

System 2:

```
/MEMORY
SIZE=512,TYPE=S,CLASS=S                      4MB OF NON-SHARED
                                             MEMORY
SIZE=64,TYPE=S,CLASS=S,MULTI                 128KW TO BE SHARED
SIZE=112,TYPE=N,CLASS=S,MULTI                224KW TO BE USED BY
                                             SYSTEM 1
SIZE=80,TYPE=S,CLASS=S,MULTI                 160KW TO BE USED BY
                                             SYSTEM 2
.
.
/PARTITION
NAME=GLOBAL01,SIZE=256,STRTPG=100,MAP=512
.
.
```

7.6.83.3 Shadow Memory Error Messages

If shadow memory is not configured properly, the following SYSGEN directive error message is displayed:

```
*** SHADOW MEMORY CONFIGURATION VIOLATION
```

If RESERVED is used with nonshadow memory types, the following SYSGEN directive error message is displayed:

```
*** ONLY SHADOW MEMORY CAN BE RESERVED
```

For more information on shadow memory, refer to Chapter 10 of this volume.

7.6.83.4 Shared Memory Error Messages

Note: A system can have only one type of shared memory configured.

If CACHE=N is specified for a second region of contiguous memory, the following message is displayed and SYSGEN aborts:

```
*** ONLY ONE CONTIGUOUS UNCACHED REGION ALLOWED
```

If INIT or RMS options are used without SHARED=Y, the following message is displayed and SYSGEN aborts:

```
*** INVALID KEYWORD WHEN SHARED=Y NOT SPECIFIED
```

If a SYSGEN SIZE directive specifies the MULTI option for memory discontiguous with a previous MULTI specification, SYSGEN generates a SG40 error and the following message is displayed:

```
WARNING; A CONTIGUOUS MEMORY RANGE WILL BE UNCACHED
```

7.6.84 SMD Directive

The SMD directive is included for compatibility and is ignored by SYSGEN. Items following this directive on the same line are ignored.

Syntax

SMD

7.6.85 //SOFTWARE Directive

The //SOFTWARE directive indicates the beginning of the software configuration section of directives. This directive is required.

Syntax

//SOFTWARE

7.6.86 SVC Directive

The SVC directive increases the size of the SVC type 1 table. If this directive is not specified, the default size is 7F. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

SVC=*num*

num	is the hexadecimal number, X'80' through X'FF' of entries allowed in the SVC type 1 table.

7.6.87 SWAPDEV Directive

The SWAPDEV directive specifies the default swap device. When an IPL or restart is performed, the default swap device is automatically assigned as the device on which the swap volume is mounted. This directive appears under the //HARDWARE /SYSDEVS subsection.

If the SWAPDEV directive is not specified, the system volume is the default swap volume.

Syntax

SWAPDEV={*devmnc* | **IPLDEV**}

devmnc	is the 2-character device mnemonic, the 2-digit hexadecimal channel number, and the 2-digit hexadecimal device subaddress of the device to be assigned as the swap device
IPLDEV	specifies the system volume as the swap volume

Specifying SWAPDEV=*devmnc* causes H.SINIT to mount the swap volume on the specified device during IPL or restart. If any errors occur while H.SINIT is mounting the swap device, they are handled according to whether the SYSGEN directive MODE=SNOP has been specified. (See the MODE Directive section of this chapter for more information.) If MODE=SNOP has been specified, H.SINIT will report mounting errors to the system console and make the system volume the swap volume. If MODE=SNOP has not been specified, an error message will be displayed on the system console and H.SINIT will prompt the user with the following:

```
ENTER SWAP DEVICE CHANNEL AND SUBADDRESS
      (OR <CR> IF SYSTEM VOLUME):
```

Specifying SWAPDEV=IPLDEV causes H.SINIT to make the system volume the swap volume. If an error occurs while H.SINIT is mounting the system volume, H.SINIT will abort.

If the SWAPDEV directive is not supplied in the SYSGEN directive file, H.SINIT will issue the following prompt during IPL or restart, unless the MODE=SNOP has been specified:

```
ENTER SWAP DEVICE CHANNEL AND SUBADDRESS
     (OR <CR> IF SYSTEM VOLUME):
```

This prompt will appear again if any error occurs while H.SINIT is mounting the swap volume.

If MODE=SNOP has been specified, H.SINIT will make the system volume the swap volume without issuing a prompt.

7.6.88 SWAPLIM Directive

The SWAPLIM directive specifies the minimum partial swap quantum. This directive appears under the //SOFTWARE /PARAMETERS subsection. If this directive is not specified or if 0 is specified for *n*, and if memory allows, swapper swaps on demand one map block at a time. If a number greater than 0 is supplied for *n*, swapper swaps out *n* map blocks of the task or the entire task, whichever is smaller. To achieve total task swapping for a 32/87 *n* must be at least 128. To achieve total task swapping for a 32/67 or a 32/97 use 2048 for *n*. The value specified is dependent on system use. When heavy swapping is anticipated and small tasks are running, a value between 7 and 15 is recommended.

Syntax

SWAPLIM=*n*

n is the minimum partial swap quantum in map blocks

7.6.89 SWAPSIZE Directive

The SWAPSIZE directive specifies the initial swap file size. If this directive is not specified, the default is 2 times the configured memory. If 0 is specified, swapping is inhibited. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

SWAPSIZE=*size*

size is the initial swap file size in megabytes

7.6.90 SWP Directive

The SWP directive is included for compatibility and is ignored by SYSGEN. Items following this directive on the same line are ignored.

Syntax

SWP

7.6.91 SYCSIZE Directive

The SYCSIZE directive is included for compatibility only. This directive is accepted by SYSGEN, but the results are not used in job processing. Default is 100 blocks. **.BB This directive appears under the //SOFTWARE /FILES subsection.

Syntax

SYCSIZE=*blocks*

blocks is the number of 192-word blocks of disk space to be allocated for each SYC file

7.6.92 SYMTAB Directive

The SYMTAB directive names the permanent file where the symbol table of the generated system is written. If the file does not currently exist, SYSGEN creates the file. This directive is required. It appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

SYMTAB=*filename*

filename is the 1- to 8-character ASCII file name of the system symbol table file

7.6.93 /SYSDEVS Directive

The /SYSDEVS directive designates the system device subsection of the //HARDWARE section. This directive is required.

Syntax

/SYSDEVS

7.6.94 SYSMOD Directive

The SYSMOD directive replaces system modules with other modules or removes the compatibilty mode system modules H.ALOC, H.FISE, H.MONS, and H.CALM. One SYSMOD directive is required for each system module to be replaced.

This directive appears under the //SOFTWARE /OVERRIDE subsection.

Syntax

SYSMOD=*name1***,REPMOD=***name2*

name1 is the 1- to 8-character ASCII name of the system module to be replaced or removed

name2 is the 1- to 8-character ASCII name of the replacement module or NULL

7.6.95 SYSONLY Directive

The SYSONLY directive specifies SYSTEM as the only valid ownername. This directive appears under the //SOFTWARE /SECURITY subsection. If an M.KEY file exists, this directive is ignored. For more information, refer to Chapter 10, the System Administrator Services chapter, in this volume.

Syntax

SYSONLY

7.6.96 SYSTEM Directive

The SYSTEM directive names the permanent file where the generated system is written. If the file does not currently exist, SYSGEN creates the file as a system file.

This directive is required. It appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

SYSTEM=*sysfile*

sysfile is the 1- to 8-character name of the file to contain the resident system image generated at SYSGEN. This file name must not be the same as the name of the current default image.

7.6.97 SYSTRAP Directive

The SYSTRAP directive selectively overrides any of the default trap handlers listed under the PROGRAM directive. This directive appears under the //HARDWARE /TRAPS subsection.

The SYSTRAP directive cannot be used if a PROGRAM directive is used.

Syntax

SYSTRAP=*name1***,REPTRAP=***name2*

name1 is the default trap handler name to be overridden (e.g., H.IP10)

name2 is the 1- to 8-character ASCII program name assigned to LFC OBJ used in place of *name1*

7.6.98 /TABLES Directive

The /TABLES directive designates the tables subsection of the //SOFTWARE directives. This directive is required.

Syntax

/TABLES

7.6.99 TERMPRI Directive

The TERMPRI directive specifies the execution priority level for all tasks activated in the interactive terminal environment. If this directive is not specified, the default is priority 60. This directive appears under the //SOFTWARE /PARAMETERS subsection.

Syntax

TERMPRI=*nn*

nn is the 2-digit decimal time-distribution priority level, 55 to 64, for interactive processing.

7.6.100 TIMER Directive

The TIMER directive specifies the number of timer table entries to be generated in the MPX-32 resident image. If this directive is not specified, the default value is 0. This directive appears under the //SOFTWARE /TABLES subsection.

Syntax

TIMER=*number*

number is the number of timer table entries to be generated

7.6.101 TITLE Directive

The TITLE directive attaches identifying information to the SYSGEN listed output. If this directive is specified, it must be the first directive in the directive stream.

Syntax

TITLE=*data*

data is 1 to 66 ASCII characters of information

7.6.102 TQFULL Directive

The TQFULL directive specifies the maximum time quantum that a time-distribution task can acquire prior to preemption by another time-distribution task at the same priority level. This directive appears under the //SOFTWARE /PARAMETERS subsection. If TQFULL or TQMIN is supplied, then both are required. TQFULL must be greater than TQMIN. If not specified, TQFULL defaults to 600 milliseconds.

Syntax

TQFULL=*time*

time is the maximum number of milliseconds in the quantum

7.6.103 TQMIN Directive

The TQMIN directive specifies the minimum time quantum that a time-distribution task can acquire prior to preemption by another time-distribution task at a higher priority level. This directive appears under the //SOFTWARE /PARAMETERS subsection. If TQMIN or TQFULL is supplied, then both are required. TQMIN must be less than TQFULL. If not specified, TQMIN defaults to 200 milliseconds.

Syntax

TQMIN=*time*

time is the minimum number of milliseconds in the time quantum

7.6.104 TRACE Directive

The TRACE directive initializes the system trace flag word C.TRACE. This directive appears under the //SOFTWARE /PARAMETERS subsection. Bit indicators within C.TRACE are described in the MPX-32 Technical Manual, Volume I, Chapter 6.

If this directive is not specified, C.TRACE defaults to X'FFFFFFFE'.

Syntax

TRACE=*num*

num is a 1- to 8-character hexadecimal number

If the TRACE directive is specified, the following NAME directive must be specified under the //SOFTWARE /PARTITION subsection:

`NAME=`*name*`,SIZE=16,STRTPG=3EF,MAP=60`

name is the 1- to 8- character user-specified name of the trace partition

7.6.105 /TRAPS Directive

The /TRAPS directive designates the trap descriptor subsection of the //HARDWARE section. This directive is required.

Syntax

/TRAPS

7.6.106 TSMEXIT Directive

The TSMEXIT directive allows TSM to exit from the system when it is no longer required. TSM will be reactivated when one of the following occurs:

- a wake up character is entered
- a task performs an M.BATCH
- a task performs an M.DEFT

If this directive is specified, it appears under the //SOFTWARE /PARAMETERS subsection.

To disable exit of TSM, see the NOTSMEXIT Directive.

Syntax

TSMEXIT

7.6.107 USERPROG Directive

The USERPROG directive specifies system modules to be included in the system along with the default trap handlers or those specified in the PROGRAM directive. This directive is required to include the H.ADA module. This directive appears under the //HARDWARE /TRAPS subsection.

Syntax

USERPROG=(*name1*, . . . *name7***)**

name1,...name 7
: are 1- to 8-character ASCII program names of modules assigned to LFC OBJ, separated by commas. A maximum of 7 names can be entered per directive.

7.6.108 /VP Directive

The /VP directive designates the Vector Processor (VP) subsection of the //HARDWARE section. This directive is required for systems containing a VP3300 or VP6410.

Syntax

/VP

7.6.109 VP Directive

The VP directive specifies the device characteristics of the Vector Processors to be configured. This directive is required for systems containing a VP3300 or VP6410 VP. This directive appears under the //HARDWARE /VP subsection.

Syntax

VP= {(*aa* [,*number*]) | *aa*} [,**PROGRAM=***module1*]

aa — is the 2-digit hexadecimal subchannel number

number — is the number of VPs to be configured. If not specified, the default is 1.

module1 — is the name of the module used to include a VP memory partition into a task's logical address space. If not specified, the default is H.ACBA.

7.6.110 VPID Directive

The VPID directive specifies unit specific information for a VP. This directive is required for systems containing a VP3300 or VP6410. One VPID directive must be used for each Vector Processor specified in the VP directive. This directive appears under the //HARDWARE /VP subsection.

Syntax

VPID=*aa*, **VPTYPE=***tt*, **STARTBLK=***blk*, **PRIORITY=***intlev*, **INTRPT=(***cc,ss***)**
[,**PROGRAM=***module2*] [,**IPCA=***ipsize*] [,**BUS0=***b0size*] [,**BUS1=***b1size*]
[,**BUS2=***b2size*] [,**BUS3=***b3size*]

aa — is the 2-digit hexadecimal subchannel number

tt — is the type of VP: 33 specifies a VP3300, and 64 specifies a VP6410

blk — is the decimal starting physical map block address where the VP memory partitions are to be allocated

intlev — is the 2-digit hexadecimal interrupt level

cc — is the RTOM/IOP channel address

ss — is the RTOM/IOP 2-digit hexadecimal interrupt subaddress

[,**PROGRAM=***module2*]
module2 is the name of the interrupt handler to be used with this VP. If not specified, the default is H.IPVP.

[,**IPCA=***ipsize*]
ipsize is the number of pages to be allocated for the Interrupt Processor Context. If not specified, the default is 16 pages.

[,**BUS0=***b0size*]
b0size is the number of pages allocated for BUS0. If not specified, the default is 16 pages. This parameter is valid for VP6410 only.

[,**BUS1=***b1size*]
b1size is the number of pages allocated for BUS1. If not specified, the default is 32 pages for a VP6410 and 48 pages for a VP3300.

[,**BUS2=***b2size*]
b2size is the number of pages allocated for BUS2. If not specified, the default is no pages. This parameter is valid for VP3300 only.

[,**BUS3=***b3size*]
b3size is the number of pages allocated for BUS3. If not specified, the default is no pages. This parameter is valid for VP3300 only.

8 System Debugger

8.1 Introduction

The System Debugger is provided on the Master SDT. It debugs the resident operating system as well as SYSGENed user interrupt and I/O handlers. The debugger is SYSGENed as part of the resident MPX-32 starter system, and it can be included in any user configuration of the resident system by using the SYSGEN directive USERPROG=DEBUG.

The System Debugger is composed of two parts: a small portion runs mapped as part of the resident operating system and occupies approximately 600 words of system logical address space; the remaining portion runs unmapped and represents the bulk of the debugger. The unmapped portion permanently occupies approximately 8KW of physical memory, but does not increase the size of the logical address space devoted to the operating system.

The system debugger only operates in privileged mode because it is using stand-alone I/O. Therefore, even privileged instructions are executed. Invalid instructions are flagged with an asterisk immediately following the opcode text. The instructions that may alter the program counter in the pseudo-PSD are executed, such as LPSD, BRI, SVC, etc., and traced by the debugger.

Unusual instruction sequences (branch increment with positive register contents, double word instructions using an odd number register, etc.) are flagged with a question mark following the instruction text. These inform the user of possible problems.

The debugger always starts in symbolic mode to allow addressing by base name plus offset. The AB directive switches to absolute addressing. The SY directive switches back to symbolic addressing. (Absolute or symbolic modes apply to all address displays.)

Each time the debugger is entered, C.GINT (the global interrupt counter) is incremented by 1. When the debugger is exited, C.GINT is decremented by 1. This defers all task scheduling during the debug session. To allow a schedule event to occur while debugging, use the CT directive specifying the address of the next instruction to be executed.

Physical and virtual (logical) modes are maintained by the System Debugger. When entry into the debugger is made from an unmapped state, the physical mode is the default. When entry is made from a mapped state, the virtual mode is the default. Execution traces and pure memory references are relative to the default mode established on entry. Symbolic references to memory toggle the display mode to that of the base symbol used; however, the execution trace mode remains in the mode of entry. The BA directive can be used to define or redefine the attributes of any base. Execution tracing is not permitted within a mapped out task's virtual address space.

8.2 Using the Debugger

8.2.1 Arithmetic and Special Operators

The debugger recognizes the following characters as unique operators allowable in any command or expression:

Character	Usage
G thru Z	symbolic base values
R	register designator
parenthesis ( )	contents of
Colon (:)	current value of last displayed contents
Asterisk (*)	indirect address if first character in a field
Slash (/)	current total
Semicolon (;)	task high address
Question Mark (?)	task low address
Dollar Sign ($)	current PC contents from PSD
Plus Sign (+)	field on left added to field on right
Minus Sign (-)	field on right subtracted from field on left
Asterisk (*)	field on left multiplied by field on right
Slash (/)	field on left divided by field on right
Ampersand (&)	field on left is ORed with field on right
At-sign (@)	field on left is ANDed with field on right
Right Angle Bracket (>)	field on left is shifted right by the count in the right side field
Left Angle Bracket (<)	field on left is shifted left by the count in the right side field
Apostrophe (')	ASCII text delimiter
Backslash (\)	subfield delimiter

8.2.2 Special Functions

The debugger expects to read a 2-character directive and options; however, the following special functions may be input instead of the normal directives:

Character	Usage
^	display previous location
*	display location indirect to current location
<ret>	display next location
>	right shift current location and display
<	left shift current location and display
$	display current location

These functions require only the one byte of input.

8.2.3 Execution Breakpoints

A breakpoint is an address where program execution can be interrupted. Eight fixed breakpoints can be defined. In addition, a one-shot breakpoint can be established for the directed execution directives GO or CT.

8.2.4 Debugger Bases

The debugger uses a range of 1-character bases to provide symbolic references to memory. Once established, the base characters (characters G through Z) can be used in any directive or arithmetic expression. The following bases are initialized by the debugger when loaded with MPX-32:

Base	Module
G	H.TAMM
H	H.MEMM
I	H.EXEC
J	available
K	available
L	H.ALOC
M	H.FISE
N	logical starting address of current task's TSA
O	available
P	available
Q	IP06 (SVC)
R	H.SWAPR
S	X'78000' (User DEBUG)
T	H.TSM
U	logical starting address of current task's DSECT
V	H.VOMM
W	H.MONS
X	H.REXS
Y	H.IOCS
Z	H.REMM

Note: 40I is equivalent to 40+I or I+40.

8.2.5 Base Characters

The characters 'G' through 'Z' can be set to any value by the BA (Base) directive. For example,

```
BA G 427F0
```

defines `G` as location `427F0`. G appears in all address displays in place of 427F0. G can be used in any directive and it is processed as 427F0.

All predefined bases except N and U have the physical attribute by default; N and U have the virtual (logical) attribute by default. These attributes are in effect until redefined using the BA directive. Virtual addresses are established using the map registers of the current task executing within the CPU. Therefore, if no task is current, the virtual attribute is undefined at that instance. Extended MPX-32 module bases are virtual when a task is current.

8.2.6 Operator Restrictions

Arithmetic expressions are evaluated left to right.

Operators can appear in any directive and are not restricted in length. For example:

```
AR D9C35FFF,@3800000,>14
```

This results in a hexadecimal 18.

```
AR *34595,<2, +Z, +3000
```

If location 34595 contained 500 (hex) and Z was set to 30000, the above expression has a value of 34400.

```
DM O + 34C2 -1>2
```

The location specified by base O was previously set to 30000, so the DM location is CD30 (30000 + 34C2 -1 and right-shifted 2 bits).

The above examples illustrate expressions that can be used in any appropriate directive.

8.2.7 Expressions

The debugger processes expressions from left to right. For example,

```
AR 2*32+G/2
```

breaks down to 2 times 32 plus the value of G. That total is divided by 2 and the answer is typed.

8.2.8 Registers

The contents of a register is specified in any expression by enclosing the register number in parentheses. For example,

```
DM (1)
```

displays the contents of R1.

The DR directive displays registers and the CR directive can modify the registers.

8.2.9 Indirection

Indirection must precede a field. For example,

```
DM *0
```

causes the value at location 0 to be obtained and displayed as an address.

8.3 Accessing the Debugger

If the system debugger is configured as part of a resident system, the debugger is automatically accessed by SYSINIT when the configured system is installed from the User SDT. (The installation process is described in Chapter 2 of this volume.) The debugger is activated by SYSINIT. The debugger input prompt is a double angle bracket (>>).

Debugger directives can be issued from the system console and patches can be made in the memory resident image of the system. The debugger is terminated by a TE (Terminate) directive. It returns control to SYSINIT, which continues building the system as described in Chapter 2.

There are three other ways the debugger can be accessed:

- A privileged user task can code a branch and link through the communications region variable that points to the debugger (C.DEBUG):

  ```
  BL *C.DEBUG
  ```

  Branch and link can only be used in modules or tasks that do not reside in the extended execution area of MPX-32.
- For modules that reside in the extended execution area or vary between extended and nonextended placement, the MBR_DBG macro can be used to access the debugger. The MBR_DBG macro determines the operating mode of the calling module and inserts the appropriate code for debugger access.
- An OPCOM DEBUG directive can be used.

The system debugger uses stand alone drivers to perform I/O. The debugger routes listings to the printer configured as LP7E and it gets directives from the terminal or teletype configured as TY7E (usually the system console). Although any terminal user can issue the OPCOM DEBUG directive, once the system debugger gains control, its prompt is displayed on the system console and it accepts directives only from that device.

8.4 Debugger Directives

When the debugger is entered, it initializes bases and sets symbolic mode. It is then ready to receive and process directives.

The directive syntax requires the first two characters be entered unless special operators are used. Certain directives contain one or more fields of additional information, separated by a comma.

Debugger directives are summarized below and described in detail in the following pages. The summary is divided into three sections: general debugging directives, directive list directives, and patch list directives.

Directive	Description
AB	displays all subsequent addresses as numeric (absolute mode)
AD	displays low and high limits of a task address space
AR	evaluates an arithmetic expression and displays its value
AS	converts an instruction into its hexadecimal equivalent
BA	creates, deletes, or modifies the definition of a user base and displays its addresses
BR	sets a breakpoint at a specified address
BY or TE	exits the debugger (only when entered by a branch and link)
CB	changes the contents of a base register
CH	displays controller definition table (CDT) entries
CM	changes the contents of memory to a new value
CO	continues tracing or execution from a breakpoint set with a BR command
CR	changes the value of a user register
CT	continues processing, setting a one-shot breakpoint at a specified address which terminates the trace function
DB	displays the base registers
DE	deletes a breakpoint which was set with the BR command
DI	displays memory locations in instruction format within a specified address
DM	displays memory locations within a specified address range
DQ	displays dispatch queue entries
DR	displays all registers
DS	displays memory locations in instruction format within a specified address range

Directive	Description
DT	dumps the event trace table to the printer. This directive assumes memory partition at hex 78000 and requires reassembly of the entire resident source with the event table enabled.
DU	dumps output to the line printer
EC	echoes terminal output to the line printer
ET	places an event trace point at a specified address
GO	resumes execution of a user program at a specified address or the last known user PSW. Optionally sets a one-shot trap at a specified address.
HC	displays the state queue head cells
LB	lists all breakpoints
LP	sets line printer output mode
LT	displays a list of the current mobile event trace points
MR	allows inspection of one CPU map register pair on a 32/87
MS	modifies scratchpad locations
PS	displays last known user PSD, program counter, condition codes, and registers
PV	converts a physical address to a virtual address
RB	resets all bases to the default values
RE	remaps the debugger to the map associated with a specified dispatch queue entry number or the current program
RT	removes a mobile event trace point at a specified address
SE	compares specified words, in the range set by the SM directive, to a specified value
SM	sets the mask as a left-justified hexadecimal number for the SE directive
SP	dumps CPU scratchpad RAM locations
SY	displays subsequent addresses as displacements from bases. This is symbolic mode.
TB	displays instructions that cause TSA stack push
TE	same as BY directive
TR	traces user programs and displays each instruction after execution
TS	terminates trace initiated by the TR directive
TY	sends output to a terminal and resets echo mode
UD	displays unit definition table (UDT) directive entries
VP	converts a virtual address to a physical address

The following directives construct a list of system debug directives and execute them at specified breakpoint(s). The directive list is also executed at the users request. The BR directive executes a directive list on a breakpoint.

Directive	Description
CD	displays the contents of the directive list
CE	zeros the command list
CL	turns off the directive list building mode
CS	enters the directive list building mode
CX	executes the directive list

The following group of directives can be used during the debug phase of IPL. These directives build a patch list of debug directives to apply to the system image being IPLed. The patch list is automatically written to disk as part of the system image at the return to SYSINIT from the system debugger. Execution of the patches is automatic at the beginning of the debug phase of IPL.

Directive	Description
PD	displays the contents of the patch list
PE	zeros the patch list
PR	turns off the patch list building mode
PT	enters the patch list building mode
PX	executes the patch list

8.4.1 AB (Absolute) Directive

The AB directive displays all subsequent addresses as numeric (absolute mode).

Syntax

AB

8.4.2 AD (Address) Directive

The AD directive displays the low and high limits of a task's address space.

Syntax

AD

8.4.3 AR (Arithmetic) Directive

The AR directive evaluates an arithmetic expression. The debugger processes the expression in its left to right order and displays the value.

Syntax

AR *expr*

expr	is the arithmetic expression to be evaluated

8.4.4 AS (Assemble Instruction) Directive

The AS directive converts an assembly language instruction to its 8-digit hexadecimal equivalent. Instruction groups supported are memory-reference and register-register. I/O instruction group is not supported. Also, the Assembler mnemonics used in the syntax for this directive are abbreviated to only 4 characters rather than 5 characters. Use the DI directive to display the valid 4-character mnemonics available for use with the AS directive.

Syntax

AS [**B**] *opcode* [,*reg*] [,*offset*] [,*index*] [(*Breg*)]

B	activates base mode operation
opcode	is the 4-character Assembler mnemonic
[,*reg*]	is a number from 0 to 7
[,*offset*]	is a hexadecimal number or an expression
[,*index*]	is the number 1, 2 or 3
[(*Breg*)]	is the base register (B0-B7) that is used for base mode instructions. If this parameter is specified, base mode must be activated (B parameter).

8.4.5 BA (Base) Directive

The BA directive defines a user base by adding its name to the internal base definition table, deletes a user base name from the base table, or redefines a user base by changing the value specified in the base name's definition. The new base possesses the physical or virtual attribute associated with the *addr* field unless overridden with the P or V options.

A maximum of 20 bases are definable.

Syntax

BA *base* [*addr*] [,**P** | ,**V**]

base	is a 1-character alphanumeric base name.
[*addr*]	is the logical or physical address for the base dependent upon the virtual or physical context of *addr*. If not supplied, the specified base name is deleted. If *addr* is supplied and *base* is already defined, *base* is redefined to represent *addr*.
[,**P**]	overrides the context for *addr* and specifies the physical attribute for *base*
[,**V**]	overrides the context for *addr* and specifies the virtual attribute for *base*

8.4.6 BR (Breakpoint) Directive

The BR directive sets a breakpoint at a specified address. A breakpoint remains in effect until cleared with the DE directive. Upon execution of the breakpoint, `BRK@`*addr* is displayed along with the contents of the registers to indicate which breakpoint was executed. A maximum of 8 breakpoints can be set. Breakpoints and event trace points (see ET command) cannot be set at the same address location.

Syntax

BR *addr* [,**C**] [,**B**]

addr	is the address at which the breakpoint is set
[,**C**]	specifies execution of a debug directive list at this address
[,**B**]	specifies that the breakpoint resides in extended MPX-32

8.4.7 BY (Bye) Directive

The BY directive exits the debugger when entry was made using a branch and link instruction. The debugger returns control to the calling program. See also the TE directive.

Syntax

BY

8.4.8 CB (Change Base Register) Directive

The CB directive modifies the contents of one or more user base registers.

Syntax

CB *reg*, *value* [,*value*]...

reg	is a user register, B0-B7
value	is the 32-bit value to be stored in the specified register. Successive values are stored in consecutive user registers. Two successive commas with no intervening value skips the user register corresponding to the missing value, leaving its contents unchanged. If B7 has been altered or skipped and an unused value remains, it is ignored.

8.4.9 CD (Display Command List) Directive

The CD directive displays the contents of the system debugger directive list.

Syntax

CD

8.4.10 CE (Zero Command List) Directive

The CE directive zeros the system debugger directive list. Once zeroed, the list can be replaced or left zeroed to remove the directive list function.

Syntax

CE

8.4.11 CH (Display Controller Definition Table Entry) Directive

The CH directive displays one or all CDT entries.

Syntax

CH [*index*]

[*index*]	is the index of the CDT entry to be displayed. If not specified, all CDT entries are displayed.

8.4.12 CL (Terminate Build Directive List Mode) Directive

The CL directive terminates the system debugger directive list building mode.

Syntax

CL

8.4.13 CM (Change Memory) Directive

The CM directive changes the contents of one or more words to a new 32-bit value. The specified address is changed to either a right-justified hexadecimal value or a left-justified blank-filled ASCII text word.

Syntax

CM *addr,value* [,*value*]...

addr	is the address of the first or only word to be changed
value	is the 32-bit value to be stored at the specified address. Successive values are stored in consecutive words beginning at *addr*. Two consecutive commas with no intervening value can be used to skip the memory address corresponding to the missing value, leaving its contents unchanged.

8.4.14 CO (Continue) Directive

The CO directive continues tracing or execution from a breakpoint. If the user is in trace mode (entered by a TR directive), tracing continues and the debugger does not exit while TR is active. If the user is not in trace mode, execution continues. (See GO directive).

Syntax

CO [*addr* [,*stop*]]

addr	is the address where program execution with optional trace continues. If not specified, the program resumes from the last known location.
stop	is the address where execution and tracing terminates

8.4.15 CR (Change Register) Directive

The CR directive modifies the contents of one or more user registers.

Syntax

CR *reg,value* [,*value*]...

reg	is a user register, R0-R7
value	is the 32-bit value to be stored in the specified register. Successive values are stored in consecutive user registers. Two consecutive commas with no intervening value skips the user register corresponding to the missing value, leaving its contents unchanged. If user R7 has been altered or skipped and an unused value remains, it is ignored.

8.4.16 CS (Build Directive List) Directive

The CS directive puts the system debugger in the directive list build mode. The following directives are not valid once CS is specified: CD, CE, CL, CX, or another CS directive.

Syntax

CS

8.4.17 CT (Continue then Terminate) Directive

The CT directive continues processing from a specified dollar sign ($) location, setting a one-shot breakpoint at a specified address which terminates the trace function.

Syntax

CT [*addr*]

[*addr*]	is the address where an optional one-shot breakpoint is set. Tracing is terminated and the users program re-entered. If not specified, return to the user's program is at the last known program counter value.

8.4.18 CX (Execute Directive List) Directive

The CX directive executes the system debugger directive list.

Syntax

CX

8.4.19 DB (Display Base Register) Directive

The DB directive displays one or more user base registers.

Syntax

DB [*Breg*]

[*Breg*]	is a user base register, 0-7. If no register is specified, the default is all user base registers are displayed.

8.4.20 DE (Delete) Directive

The DE directive deletes a breakpoint that was set with a BR directive and restores user instructions to their original locations.

Syntax

DE *addr*

addr	is the address where the breakpoint is deleted and the user's instruction is restored to its original location

8.4.21 DI (Display Instruction) Directive

The DI directive displays any memory locations within a specified address range, in instruction format, one line at a time (same as DS directive).

Syntax

DI *add1* [,*add2*] [,**B**]

add1	is the address where the display starts
[,*add2*]	is the address where the display ends. If not specified, the display continues until a character other than a carriage return is entered at the end of a line.
[,**B**]	specifies base mode instruction display

8.4.22 DM (Display Memory) Directive

The DM directive displays all memory locations within the specified address range. A carriage return continues the display; any other character entered terminates the display.

Syntax

DM [*add1* [,*add2*] [,*number*]]

add1	is the address where the display starts
[,*add2*]	is the address where the display ends. If not specified, the ending address defaults to the starting address. For example, *add1* is the only location displayed.
[,*number*]	is the number of words to be displayed per line on the console (minimum of 1, maximum of 8). If not specified, the default is 4.

8.4.23 DQ (Display Dispatch Queue Entry) Directive

The DQ directive displays one or all DQE entries.

Syntax

DQ [*index*]

[*index*]	is the index of the DQE entry to be displayed. If not specified, all DQE entries are displayed.

8.4.24 DR (Display Register) Directive

The DR directive displays the memory locations of registers.

Syntax

DR [*reg*]

[*reg*]	is a user register (R0-R7). If not specified, all registers are displayed.

8.4.25 DS (Display Symbolic) Directive

The DS directive displays any memory locations within a specified address range, in instruction format, one line at a time (same as DI directive).

Syntax

DS *add1*[,*add2*] [,**B**]

add1	is the address where the display starts
[,*add2*]	is the address where the display ends. If not specified, the display continues until a character other than a carriage return is entered at the end of a line.
[,**B**]	specifies base mode instruction display

8.4.26 DT (Display Event Trace) Directive

The DT directive dumps the event trace table to the line printer. Event trace is disabled until printing is completed.

Notes:

This directive assumes memory partition at hexadecimal 78000 and requires reassembly of the entire resident source with event trace enabled.

Syntax

DT

8.4.27 DU (Dump) Directive

The DU directive writes a range of memory to a line printer. ASCII format is used for the right-hand side of the memory display.

Syntax

DU [*start* [,*stop*]]

start	is the memory address where the dump starts
stop	is the memory address where the dump stops. If not specified, the default is the end of the operating system.

If no parameters are specified, the entire operating system area is output.

8.4.28 EC (Echo) Directive

The EC directive generates hard copy output when a CRT is the terminal device. Output is listed, line by line, on the line printer for every carriage return on the terminal. To terminate the EC directive, specify the TY directive.

Syntax

EC

8.4.29 ET (Enter Event Trace Point) Directive

The ET directive places an event trace point within either resident or nonresident code without requiring reassembly of a program. To turn on the trace, this directive resets bits 21 and 22 of C.TRACE.

Prerequisites are:

1. The system must have been SYSGENed with a static partition for the event trace table.
2. Both H.DBUG (System Debugger) and H.IP06 (SVC Trap Handler) must have been assembled with TRACE set true.

A maximum of 8 event trace points can be set. Event trace points and breakpoints (see BR directive) cannot be set at the same address location.

Syntax

ET *addr* [,**B**]

addr	is an absolute or relative address with a symbolic base
[,**B**]	specifies that the event trace point resides in extended MPX-32

8.4.30 GO (Go) Directive

The GO directive transfers control to the user task, optionally setting a one-shot trap at a specified address.

Syntax

GO [*start* [,*stop*]] [,**B**]

[*start*]	*start* is the address where the program starts executing
[,*stop*]	*stop* is the address where the program stops executing. When specified, a breakpoint is set at this address before execution begins at the *start* address. If not specified, the program is reactivated at the *start* address.
[,**B**]	specifies base mode execution

If no parameters are specified, the program is entered at the last known user PSW.

8.4.31 HC (Display Dispatch Queue Head Cell) Directive

The HC directive produces a 3-column display of the MPX-32 state queue head cells. Each entry is 3 words long.

Syntax

HC

8.4.32 LB (List Breakpoint) Directive

The LB directive lists all active fixed breakpoints set using a BR directive.

Syntax

LB

8.4.33 LP (Line Printer) Directive

The LP directive directs output to the line printer. If the user is tracing, no request is made for input after each instruction. A breakpoint must be set to terminate the trace.

Syntax

LP

8.4.34 LT (List Mobile Event Trace Point) Directive

The LT directive lists current mobile event trace points within resident or non-resident code.

Prerequisites are:

1. The system must have been SYSGENed with a static partition for the event trace table.
2. Both H.DBUG (system debugger), and H.IP06 (SVC trap handler), must have been assembled with TRACE set true.

Syntax

LT

8.4.35 MR (Map Register) Directive

The MR directive allows inspection of one CPU map register pair. This directive is available only on the 32/87 computer.

Syntax

MR *index*

index is the number of the map register from 0 to FF

8.4.36 MS (Modify CPU Scratchpad Location) Directive

The MS directive modifies any location within the CPU scratchpad random access memory. Because the scratchpad is used by the CPU to process I/O and interrupt requests, incorrect modification of scratchpad locations could cause undesirable results. This directive should be used with extreme caution.

The scratchpad address specified in this directive can be within two ranges. The first range is a number between 0 and FF corresponding to the actual scratchpad address as defined in hardware. The second range is a number between 300 and 6FC corresponding to a scratchpad location obtained by using the SP directive to display all scratchpad locations.

If an invalid address is specified or an attempt is made to write past the end of scratchpad, an error message is displayed.

Syntax

MS *loc,value* [,*value*]...

loc	is the scratchpad address 0 to FF, or 300 to 6FC
value	is the 32-bit value stored in the specified location. Successive values are stored in successive locations.

8.4.37 PD (Display Patch List) Directive

The PD directive displays the contents of the system debugger patch list. The display list is either the previous patch list written to disk or the current edited version.

Syntax

PD

8.4.38 PE (Zero Patch List) Directive

The PE directive zeros the system debugger patch list. Once zeroed, the list can be replaced or left zeroed to remove the patch function.

Syntax

PE

8.4.39 PR (Terminate Build Patch List Mode) Directive

The PR directive terminates the system debugger patch list building mode.

Syntax

PR

8.4.40 PS (Program Status) Directive

The PS directive displays the user program status doubleword (PSD), program counter, condition codes, and registers.

Syntax

PS

8.4.41 PT (Build Patch List) Directive

The PT directive causes the system debugger to enter the patch list build mode. The following directives are not valid after PT has been specified: PD, PE, PR, PX, or another PT directive.

Syntax

PT

8.4.42 PV (Convert Physical Address to Virtual Address) Directive

The PV directive converts a physical address to a virtual address.

Syntax

PV *addr*

addr is the physical address to be converted to a virtual address

8.4.43 PX (Execute Patch List) Directive

The PX directive executes the system debugger patch list.

Syntax

PX

8.4.44 RB (Reset Bases) Directive

The RB directive resets all bases to their initial default values.

Syntax

RB

8.4.45 RE (Remap) Directive

The RE directive remaps the debugger to the map associated with a program.

Syntax

RE [*number*]

[*number*]	is the dispatch queue number in the range 0 to FF where remapping is associated. If not specified, remapping defaults to the dispatch queue number in C.CURR.

8.4.46 RT (Remove Event Trace Point) Directive

The RT directive removes a mobile event trace point within resident or nonresident code.

Prerequisites are:

1. The system must have been SYSGENed with a static partition for the event trace table.
2. Both H.DBUG (system debugger), and H.IP06 (SVC trap handler), must have been assembled with TRACE set true.

Syntax

RT *addr*

addr	is an absolute or relative address with a symbolic base

8.4.47 SE (Search Equivalent) Directive

The SE directive compares specified words to a specified value. Each word is ANDed with the search mask (see SM directive) before being compared to the value. Each bit in the range is listed.

Syntax

SE *value, start, stop*

value	is the value where each word is compared
start	is the address where the search begins
stop	is the address where the search ends

8.4.48 SM (Set Mask) Directive

The SM directive sets the mask for the SE directive. The mask parameter is interpreted as a left-justified hexadecimal number or right-justified ASCII character string.

Syntax

SM [*mask*]

mask	is a new mask value. If not specified, the default is the previously entered value; if none, X'FFFFFFFF' is used.

8.4.49 SP (Scratchpad Dump) Directive

The SP directive outputs to the terminal the contents of the scratchpad locations (23 lines of text are output at a time). A carriage return continues output, entering any other character terminates output.

Syntax

SP [*loc*]

[*loc*]	is the CPU scratchpad address 0 to FF. If not specified, all scratchpad locations are displayed.

8.4.50 SY (Symbolic) Directive

The SY directive displays all subsequent addresses as displacements from bases (symbolic mode).

Syntax

SY

8.4.51 TB (Trace Back) Directive

The TB directive displays the instructions that caused each TSA stack push. TB is only valid if there is a current task which is not at the base level of its stack. Make sure the debugger is mapped with the correct task space. The instructions are displayed in logical order. For example, the first instruction shown is the instruction executed by the task (usually a SVC) that caused the first push. Because some operating system routines allocate extra stack frames as work space, some displays can contain apparently invalid information.

Syntax

TB

8.4.52 TE (Terminate) Directive

The TE directive exits the debugger if it was entered using a branch and link. Control returns to the calling program. See the BY directive.

The TE directive should not exit from a breakpoint. See the CO and CT directives.

Syntax

TE

8.4.53 TR (Trace) Directive

The TR directive executes and displays results of user instructions one at a time. Addresses are displayed as a base character plus offset value.

The last instruction executed is displayed and the cursor is held at the end of the line awaiting a user directive. A carriage return or line feed causes the next instruction to be executed and displayed. An up arrow (^) causes the previous instruction to be redisplayed. An equal sign (=) causes the hexadecimal equivalent of the instruction just executed to be displayed. Any other character causes the debugger to prompt for a directive.

Symbols established by the BA directive are used for display purposes if the SY directive was set.

Execution tracing is not permitted within a mapped out task's virtual address space.

Syntax

TR [*start* [,*stop*]] [,**B**]

[*start* [,*stop*]]

start is the address of the first user instruction to be traced. If not specified, the default is $ (current PSD value).

stop is the address of the last user instruction to be traced. If not specified, tracing continues without bounds.

[,**B**] specifies base mode display and execution for tracing through extended MPX-32.

8.4.54 TS (Trace Stop) Directive

The TS directive exits the trace mode initiated by the TR directive. All further I/O is directed to the user's terminal.

Syntax

TS

8.4.55 TY (Terminal) Directive

The TY directive directs output to a terminal and to reset EC (Echo) mode.

Syntax

TY

8.4.56 UD (Display Unit Definition Table Entry) Directive

The UD directive displays one or all UDT entries.

Syntax

UD [*index*]

[*index*] is the index of the UDT entry to be displayed. If not specified, all UDT entries are displayed.

8.4.57 VP (Convert Virtual Address to Physical Address) Directive

The VP directive converts a virtual address to a physical address.

Syntax

VP *addr*

addr is the virtual address to be converted to a physical address

8.5 System Debugger Practice Debug Session

The following example is for first time users of the system debugger. It is recommended that the user try the following directives on the system console of an MPX-32 system. This should only be done on a stand-alone system (with no other users on the system), because it is very easy for the inexperienced user to crash the system.

This example does not attempt to explain every directive or even the complete syntax of the directives. It is intended to help use the system debugger. While using this example, please refer to the directives descriptions and read each description before proceeding.

8.5.1 Step One - Accessing the Debugger

The debugger is generally entered using the instruction:

```
BL *C.DEBUG
```

This is the case whether it is entered by a user written program or through the OPCOM DEBUG directive. The calling program must be privileged to make this call. Also, the macro 'M.EQUS' or 'M.COMM.' must be included in the program to allow the symbol 'C.DEBUG' to be properly assembled.

To exit the debugger, when entered by a branch and link, use the TE directive.

At the operator's console, invoke the debugger using OPCOM.

TSM>OPCOM		TSM>OPCOM DEBUG		TSM>!DEBUG
??DEBUG	or	>>	or	>>
>>				

Then exit.

>>TE	>>TE	>>TE
??	TSM>	TSM>

Notice you return to the process you left.

8.5.2 Step Two - Task Debugging with the System Debugger

While it is normally best to debug a task using the task debugger, there are cases where the system debugger is the better tool, because the system debugger allows the user to trace through system calls.

8.5.3 Using the System Debugger to Display Memory

At the console, perform the following:

```
TSM>!DEBUG      Enter the debugger (OPCOM is the task)
>>
```

Because Assembler generated listings have the program relative addresses listed, it is necessary to set a base at the start of the task to be debugged. This allows the use of the offsets given by the Assembler. To do this, it is necessary to know where the task starts. This information is contained within the Task Service Area (TSA) represented by the predefined debug base N.

To set a user base, first select the character desired as the base. In this example, use `P`. Using the macro expansion of the TSA variables, the start of the task is at T.BIAS which equates to hexadecimal '704'. To set the base, use the following debug directive.

```
>>BA P *704N
```

This sets base `P` to equal the virtual address stored at hexadecimal '704' relative to the TSA. The * indicates the address specified is indirect. When referencing locations that consist of an offset and a base, as in the `704N` above, the + operator is implied.

Now, use the base to look at information contained within the task.

To find where the debug call is made, use the DI (Display Instruction) directive to list the instructions with their program relative offsets. The current task program counter is pointing one word past the call to the debugger, so the following instruction displays the debug call:

```
>>DI $-4
     P+5C40 BL 0,X1
     P+5C44 LF R0,P+20
```

The `$` indicates the current program counter location. In this case, the debugger is entered by an indexed branch to allow OPCOM to test for the presence of the debugger before attempting to branch to it.

To look at more instructions, repeated carriage returns display successive physical memory locations in instruction format.

To display noninstruction memory contents, use the DM (Display Memory) directive. This directive is the default directive. If only a number is entered, the data contained at the specified address is displayed on the terminal.

For example:

```
>>DM 50P
    P+50 46554C20 *FUL *
>>50P
    P+50 46554C20 *FUL *
```

The DM directive is not always defaultable. If the memory location to be displayed is ambiguous with a debug directive, then DM must be entered.

`>>EC04` Would be interpreted by the debugger as the ECHO directive. DM must be entered.

`>>DM EC04`

Other potentially ambiguous directives are:

- AB — set absolute addressing mode
- AD — display task address limits
- BA — define base
- DE — delete breakpoint
- EC — echo terminal output to line printer

To display the registers at any point, two directives are used. The first is the DR (Display Register) directive. DR lists any or all of the eight general purpose registers.

```
>>DR
   A1037C44 00021F8C 00000000 0000002F    *..  D.........../*
   0000000D 00000004 44544255 20202020    *........DEBU *
>>DR R5
   00000004                               *....*
```

The second directive is the PS (Program Status) directive. PS lists the program status doubleword, program counter, condition codes, and general purpose registers.

```
>>PS
   PSD=     A0037C44 80000058
   PC=      G+103C   CC= 0100
R0-7
   A1037C44 00021F8C 00000000 0000002F    *..  D.........../*
   0000000D 00000004 44544255 20202020    *........DEBU *
```

Often, calls to system services require data structure addresses be passed in registers. An example is the FCB address in almost all I/O service calls. If examination of such a structure is required, two methods can be employed.

The first is to display the registers, or a particular register using the DR directive. Then the memory locations pointed to by the register containing the data can be examined by the DM (Display Memory) directive. If the last base referenced had a physical attribute, the memory displayed is physical. If the last base had a virtual attribute, the memory displayed is logical.

The second involves using the DM directive with a special operator, the parentheses, indicating 'contents of a register'. A number in the range of 0 to 7 is enclosed in parentheses and indicates the contents of a register. To look at a 16 word FCB, use the following directive.

```
>>DM (1,(1+40
```

or

```
>>(1,(1+40
```

By using the debug directives to their maximum potential, a great deal of information about a task is made readily available to the user. Two examples of such directives follow.

1. Displaying the Dispatch Queue Entry for the Current Task

   In general, have an MPX-32 Technical Manual available so that system data structures can be analyzed. For the debugger to display the actual relative addresses displayed in the diagrams, and not the absolute addresses, set a debug base. Q is an appropriate base to use for the dispatch queue. Set the base as follows.

   ```
   >>BA Q *8E8
   ```

   Because the `*` indicates indirect, base `Q` is set to the address stored at X'`8E8`'. The number X'8E8' was determined by looking at a C-dot cross reference for C.CURR. This is the dispatch queue head cell for the current task. To examine any part of the queue entry, refer to the proper offset followed by Q.

   ```
   >>Q,20Q
     Q+0   000008E8 000008E8 3C3C3C02 17000015 *........<<<.....*
     Q+10  53595354 454D2020 4F50434F 4D202020 *SYSTEM OPCOM *
   ```

   The state queue linkage, the task number, the owner name, and the task name for the current task are shown.

2. Traversing a Linked Queue

   If the integrity of a linked queue is in doubt, the debugger easily traces through the entries as shown below:

   ```
   >>Q
     Q+0 000008E8                      *....*
   >>*
     8E8 00003D30                      *..=0*
   >>*
     Q+0 000008E8                      *....*
   ```

   Because the `*` indicates indirect, it refers to the address displayed on the previous line. In this manner, linked queues of any length can be traversed to ensure they contain all required entries, are connected, or whatever else may be desired.

8.5.4 Using Debug to Display a Program

Assume a program is failing in one particular subroutine. By using the breakpoint capabilities of debug, let the program run at machine speed until the trouble spot is encountered. To set a breakpoint, use the following directive.

>>BR *addr*

To let the program run until reaching the breakpoint, do the following:

>>TE	If the debugger was entered by `BL *C.DEBUG` (or)
>>BY	If the debugger was entered by `BL *C.DEBUG` (or)
>>CO	If the debugger was entered from a breakpoint (or)
>>CT	If the debugger was entered from a breakpoint

An explanation of the four different debug exit directives follows.

TE or BY — either of these two directives can exit the debugger when it is entered by a branch and link. These are the only directives that are acceptable. Use of the CO or CT directives produce errors later on.

CO — continues the last mode of execution. If the user was tracing, then terminated trace mode using a noncarriage return input and now wishes to resume tracing, either CO or TR accomplishes it. If the program was running normally and was stopped by a debug breakpoint, CO will continue execution of the program in run mode. This directive should be used only if the most recent entry into the debugger was through a breakpoint.

CT — continues execution of the task in run mode regardless of the previous mode (trace or run). Like the CO directive, CT should only be used when the most recent entry into the debugger was through a breakpoint.

Upon reaching the breakpoint, the following is displayed:

```
>>BR 5C48P
>>TE
  BRK P+5C48 CC 0100

R0-7
  89032C08 FFFFFFF0 00000000 0000002F *..,............/*
  0000000D 00000004 44454255 20202020 *........DEBU *
```

Note: Breakpoints cannot be set on SVC instructions. An error message is displayed should the user attempt this. The trace mode should only be used when the debugger is invoked by a breakpoint. If entry was accomplished by a branch and link, a breakpoint should be set, TE executed, and the trace started after entering the breakpoint. Breakpoints cannot be set within mapped out tasks.

When the area of code to be debugged has been reached, a helpful directive is TR (Trace). This directive executes instructions, one at a time, and displays the disassembled instruction, along with any pertinent registers or memory locations, on the console.

```
>>TR
```

After tracing several instructions, a bug is found. The instruction at *xxxx*P is:

```
LD R4,100,X2
```

what should have been there is:

```
LD R4,100,X3
```

In order to continue the debugging process, the instruction must be modified and re-executed. The CM (Change Memory) directive does this.

>>CM *addr, value*

This directive requires the correct hexadecimal value for the new instruction. The debugger assembles memory reference and register-register instructions. The result is displayed in hexadecimal. To find the value needed above use:

```
>>AS LD,4,100,3
    AE600102
```

Then to patch the bad instruction:

>>CM *xxxx*P,AE600102

and to re-execute it:

>>TR *xxxx*P

If it is desired, registers can be modified by using the CR directive.

This method for patching a program works for single line errors. That is, a bad memory address, an incorrect register, or any other error requiring modification of only one line of code. For more major problems that require the insertion of more instructions, the following technique can be employed.

Because several low memory areas (for example, X'D0' to X'FF') are not used after the system is booted, we can use these locations as a scratch area for building patches. MPX-32 has a section of memory reserved for patching by J.INIT that can build temporary debug patches. It is found by examining C.MPAC (determine actual address by assembling M.EQUS). This word contains the next available address within the MPX-32 patch area. The patch area end address is contained in C.MPAH.

To use the available memory locations, first, select the instruction where code is to be added. Using the area mentioned above, replace the instruction with a branch to X'D0'. At location X'D0', build the desired instructions ending with a branch back to the proper return location. This is generally one word past the branch to X'D0'. The instructions are, for the most part, generated by the debug AS directive, placed by the CM directive, and verified before being executed by the DI directive.

8.5.5 Summary

The best way to learn the debugger is to use it. Reread section 8.4, Debugger Directives, in this chapter for complete descriptions of the directives because the following summary does not cover all of the directives or make full use of the ones covered. Type carefully because the debugger executes exactly as requested. If you do not specify a base for an address, the debugger references the absolute location.

8.6 Example of Directive List Use

Problem:

Log contents of FCB after repeated I/O operations performed by new resident operating system module without continuous user intervention.

Solution:

Place a breakpoint at 63C relative to new module. Start with the directive list option and create a directive list to dump the FCB and continue.

Implementation:

Build Directive List

1. Access the system debugger

   ```
   TSM>!DEBUG
   ```

2. Clear the directive list

   ```
   >>CE
   ```

3. Set directive list build mode

   ```
   >>CS
   ```

4. Construct directive list to dump FCB

   ```
   >>DU (1,(1+40
   >>CO
   ```

5. Terminate directive list build mode

   ```
   >>CL
   ```

6. Display directive list

   ```
   >>CD
   ```

   Resultant CRT display:

   ```
   CL  →  DU (1,(1+40
   CL  →  CO
   ```

7. Set breakpoint

   ```
   >>BR 63CX,C
   ```

8. Return to TSM

   ```
   >>TE
   ```

8.7 Example of Patch List Use

Problem:

System is generated containing user created operating system modules. The new module contains the patchable coding error

```
LW R1,0W,R3
```

(should be)

```
LA R1,0W,R3
```

Solution:

System debugger patch list containing a Change Memory (CM) directive

Implementation:

Build Patch List

1. IPL system. When the debug prompt appears, locate erroneous instruction (for example, location 43C relative).

2. Use the Assemble directive to construct new instruction

   ```
   >>AS LA,1,0,3
   ```

   Result: 34E00000

3. Clear patch list

   ```
   >>PE
   ```

4. Set patch list build mode

   ```
   >>PT
   ```

5. Build patch list

>>CM 43C(relative),34E00000	Replace instruction at 43C relative
>>TE	Return to SYSINIT

6. Terminate patch list build mode

   ```
   >>PR
   ```

7. Display patch list

   ```
   >>PD
   ```

   Result:

   ```
   PL  →  CM 43C(relative),34E00000
   PL  →  TE
   ```

8. Execute patch list

   ```
   >>PX
   ```

   Result:

   Location 43C (relative) changed to 34E00000 and control returns to SYSINIT when the patch list is written to disk.

9 Online System Patch Facility (J.INIT)

9.1 Introduction

J.INIT provides for temporary or permanent patching of the MPX-32 resident image. In addition, J.INIT performs the Alterable Control Store/Writable Control Store (ACS/WCS) loading function. See Chapter 12 of this volume.

J.INIT also initiates mount requests for any public volumes to be mounted at IPL time. The information for these requests is provided in the system file, M.MOUNT. See section 9.7, Automatic Mounting of Public Volumes, in this chapter.

J.INIT processes patch directives from the patch file.

The patch file name is supplied in the SYSGEN PCHFILE directive. This file should contain valid patch directives as described in the Patch Directives section of this chapter. The patch file is maintained by the Text Editor and should be stored as a blocked, uncompressed file. An associated symbol table file is built by SYSGEN when the SYMTAB directive is specified. The SYSGEN PATCH directive should define the size of the patch area.

Only one patch area is used for extended, nonbase, and mapped out modules. If the extended mode is activated at SYSGEN, the patch area must be located in the first 16KW of memory. Otherwise, the patch area can be located anywhere in the resident MPX-32 system.

Patches for any modules that are located in extended memory must be constructed by using the appropriate base register instructions. J.INIT is unable to perform opcode decoding or substitution for extended modules. Memory reference instructions must be encoded with the base register field equal to 0. If these instructions are designated relocatable, J.INIT inserts the appropriate base register and offsets into the address field. J.INIT generates an error if an attempt is made to enter a relocatable base register instruction with a nonbase zero field.

Entry of a patch that invokes a branch between extended and nonbase modules is not supported. Because the adapter code addresses are not available to J.INIT, and because J.INIT cannot build adapter sequences, the user must locate the required adapter and enter the branch in absolute mode.

Patch processing terminates when the Exit (/E) directive is encountered in the patch input file. At this time, an audit trail of all patches specified is written to the SLO file. The audit trail listing is suppressed by specifying the NPR option with the option (/O) directive. The patch program accepts directives to control processing. The general format of a directive is:

/d f2 f3 f4 fn

/d is the directive name that must be followed by one or more spaces. This is called field 1 for error messages.

f2 - fn are fields containing the names, values, and special symbols processed by the directive. Fields must be separated by one or more spaces or a comma.

For a patch directive to reference an extended module's DSECT data, SYSGEN generates system symbol table entries containing the DSECT base addresses. The name for these entries in the system symbol table is the module name with the first character replaced by "$.". For example, $.REMM indicates the address of H.REMM's DSECT data. These names are used on relocatable patches that are DSECT relative.

One character module names (A,B,C,etc.) or module names with only the first character unique (AMOD, BMOD, CMOD, etc.) cause the generation of multiply defined DSECT symbol table entries ($,$MOD). This generates an error in J.INIT.

9.1.1 Dedicated Names

Dedicated names used by J.INIT are:

Dedicated Name	Description
$	equivalent to the address of the next free patch area location
R	indicates a relative address in the positive direction
-R	indicates a relative address in the negative direction
;	delimits fields to be processed and comments

9.1.2 Conventions

All field entries on patch directives must conform to the following conventions:

name	1 to 8 ASCII characters, one of which must be nonnumeric
value	1 to 8 hexadecimal digits; leading zeros need not be specified. Only whole words are generated.
address	1 to 5 hexadecimal digits; leading zeros need not be specified. Must be word resolution (bits 30-31 equal 0).
label	1 to 8 ASCII characters, one of which must be nonnumeric

9.2 Patch Directives

Patch directives are summarized below and described in detail on the following pages.

Directive	Description
/B	define a base address
/C	change the contents of a memory location
/D	define a named value
/E	exit
/G	go to the patch area from a specified memory location
/O	select patching options
/P	define a patch area
/R	return from the patch area
/F, /N, /T	process patch directives conditionally
/$	enter a value into the patch area
/;	comment only

9.2.1 /B (Define a Base Address) Directive

The /B directive allows a name to be equated to a base address. This definition is inserted in the internal symbol table and can be referenced by subsequent directives.

Syntax

/B *name address*

name is equated to *address*

/B $name_1$ $name_2$

$name_1$ is equated to the address value of $name_2$

9.2.2 /C (Change the Contents of a Memory Location) Directive

The /C directive changes the value of any location in memory.

Syntax

/C *address value*

the specified value is inserted at the specified address

/C *address value* **R**

the address field of the value parameter is added to the address parameter to form the address field of value. The specified value is inserted at the calculated address.

/C *address value* **-R**

The address field of the value parameter is subtracted from the address parameter to form the address field of value. The specified value is inserted at the calculated address.

/C *name address value*

The value of the name parameter is added to the address parameter to form the actual address. The specified value is inserted at the actual address.

/C *name address value* **R**

the specified value is inserted at the specified address, relative to name. The value of name is added to value to form the actual address.

/C *name address value* **-R**

the specified value is inserted at the specified address, relative to name. The address field of value is subtracted from name to form the address portion of value.

/C $name_1$ *address value* $name_2$

the value of $name_2$ is added to the value parameter and inserted at the actual address. The value of $name_1$ is added to the address parameter to form the actual address.

9.2.3 /D (Define a Named Value) Directive

The /D directive equates a name to the address of a value. The value is stored in the next free location of the patch area. The definition is inserted in the internal symbol table and can be referenced by subsequent directives.

Syntax

/D *name value*

name is equated to the address of *value* in the patch area

/D $name_1$ $name_2$

$name_1$ is equated to the address of the value of $name_2$ in the patch area

9.2.4 /E (Exit) Directive

The /E directive terminates directive processing.

Syntax

/E

9.2.5 /G (Go to the Patch Area from a Specified Memory Location) Directive

The /G directive inserts an unconditional branch to the patch area at any memory location. The location branched to will be the next free location of the patch area plus, optionally, an offset.

Syntax

/G *address*

an unconditional branch to the specified address of the patch area is inserted

/G *name address*

an unconditional branch to the specified address of the patch area relative to *name* is inserted. The value of *name* is added to *address* to determine the actual address of the branch in the patch area.

/G *name address value*

an unconditional branch to the patch area at *address*, relative to *name* is inserted. The branch is to the next free patch location plus value number of bytes (word resolution). The offset area is thus reserved. The basic form, with no name field, can also be used.

9.2.6 /O (Select Patch Options) Directive

The /O directive specifies options for controlling the processing of subsequent directives.

Syntax

/O $name_1$ $name_2$ $name_n$

$name_1$, $name_2$ and $name_n$ are the option names. A single /O directive can contain more than one name parameter. There can be any number of /O directives.

The available options are:

Name	Use
NAM	informs J.INIT that the definitions in the patch file should be merged into the internal symbol table. This option should be specified once per single run.
NHE	informs J.INIT not to halt if patch errors are detected. If not specified, any patch error causes a halt. Entering //RUN continues processing.
NPR	informs J.INIT that a patch listing is not to be produced.
SYM	directs J.INIT to produce a listing of the internal symbol table. This specification should not be made in the first directive of a patch deck.

9.2.7 /P (Define a Patch Area) Directive

The /P directive defines a temporary patch area or appends patches to the patch area defined by SYSGEN. It should be used during debugging.

Syntax

/P *address value*

a patch area is defined starting at location *address*, *value* (word resolution) bytes long. If the extended mode is active in SYSGEN, the address must be in the first 16KW of memory. Since no attempt is made to protect this area, it should be some area of the resident image not used during debug operations.

/P *modname*

modname signals J.INIT to determine the operating mode of a module and to generate a temporary patch area address according to the mode. This allows a patch area to be situated in the extended partition.

/P CUR

subsequent patches are added to those entered during a previous patch run

9.2.8 /R (Return from the Patch Area) Directive

The /R directive allows an unconditional branch back to the instruction plus one word produced by the last /G directive encountered. An offset can be specified to reserve a number of patch locations immediately following the branch back.

Syntax

/R

an unconditional branch is inserted to the location plus one word containing the last branch generated by a /G directive

/R *value*

an unconditional branch is inserted to the location plus one word containing the last branch generated by a /G directive *value* number of bytes of the patch area (word resolution) are reserved. This reserved area follows the generated branch.

9.2.9 /F, /T, /N (Conditional) Directives

The /F and /T directives allow skipping the processing of other directives based on whether the directive statement is true (/T) or false (/F). Processing continues when a /N statement with the matching label is encountered. This creates a general patch file that attempts to modify only those modules included in the resident image.

Syntax

/F *name label*

if *name* is false or not defined in the symbol table, discontinue directive processing until a /N directive containing *label* is encountered

/T *name label*

if *name* is true or defined in the symbol table, discontinue directive processing until a /N directive containing *label* is encountered

/T EXTDMPX *modname label*

/F EXTDMPX *modname label*

if the operation mode of *modname* is extended (/T) or nonextended (/F), directive processing is discontinued until a /N directive containing *label* is encountered

/T MAPOUT *label*

/F MAPOUT *label*

if the current image is mapped out (/T) or mapped in (/F), directive processing is discontinued until a /N directive containing *label* is encountered

/N *label*

continues directive processing

9.2.10 /$ (Enter a Value into the Patch Area) Directive

The /$ directive inserts a value into the next free location of the patch area.

Syntax

/$ *value*

value is inserted into the next free patch location

/$ *value* **R**

value is inserted, relative at the actual location. The value of $ is added to the address field of the value parameter for the actual location.

/$ *value* **-R**

the address field of *value* is subtracted from $ to form the actual address field of *value*. Value is then stored at the next free patch location.

/$ *name value*

value is inserted, relative to *name*. The value of *name* is added to the *value* and placed into the next free patch location.

9.2.11 /; (Comments) Directive

The /; directive can be included on any patch directive as a delimiter. The total directive can be designated as a comment by the use of this directive.

Syntax

/; *text*

9.3 Entry Conditions

Calling Sequence:

J.INIT is activated by SYSINIT at start-up. If the system is running, J.INIT can also be activated from TSM or OPCOM. To activate J.INIT from TSM, enter:

```
TSM>J.INIT
```

To activate from OPCOM, enter:

```
??ACTIVATE J.INIT
```

Patch processing is inhibited by setting control switch 2.

9.4 Exit Conditions

Return Sequence:

M.EXIT	Exit to MPX-32

Registers:

None

9.5 External References

Abort Cases:

Halt if error detected and the NHE option has not been specified.

Output Messages:

J.INIT produces an audit trail of all patches made unless the NPR option is specified. The information produced includes a source image of each patch, the actual location patched, the actual value stored, and the previous contents of the location. The number of remaining free patch locations is also listed.

J.INIT can include error messages along with the audit trail listing. All error messages are preceded and/or followed by asterisks (*******). Possible error messages are:

`ERROR IN PREVIOUS PATCH-FIELD-`*n*
: *n* = number of the field containing the error (/d = field 1)

`BASE TABLE OVERFLOW`
: An attempt has been made to insert too many names in the internal symbol table (limit = 215_{10}).

`PATCH AREA OVERFLOW`
: An attempt has been made to insert too many patches in the area defined during SYSGEN or on the /P directive.

`DUPLICATE NAME -` *name*
: An attempt has been made to insert the displayed name in the internal symbol table and it is currently in the table.

`END OF FILE ON` *patchfilename*
: An attempt has been made to save the source image of a patch on the patch file and it is full (limit = 900 images).

`UNABLE TO ALLOCATE` *patchfilename*
: If the patch file does not exist, J.PATCH attempts to create it. This message indicates that sufficient disk space was unavailable. 100 blocks are required.

`UNABLE TO ALLOCATE INPUT DEVICE`
: The request to allocate the patch input device has been denied.

`PATCH ERRORS DETECTED`
: Output at the end of the audit trail if any patch errors were detected. Also output to the console teletypewriter.

`EXTD/NONEXTD ADDRESS CONFLICT`
: Indicates an extended patch exceeds the first 16KW of memory or an attempt was made to: place a nonbase patch in the extended area or place an extended patch in a nonbase area.

9.6 Examples

Changing locations in a resident module

```
/O NAM                         Get module description
/C H.IOCS 2154 CA803331        See Note 1
/C H.IOCS 574 EC001003 R       See Note 2
/E                             End of patches
```

Notes:

1. Changes location `2154` of `H.IOCS` to a LI R5,X'`3331`'.
2. Changes location `574` of `H.IOCS` to a BU to location `1003` of H.IOCS.

Inserting into the patch area

```
/O NPR NHE NAM          See Note 1
/G H.REXS 100           See Note 2
/$ CB050001             See Note 3
/$ H.REXS F20005E5      See Note 4
/$ EE000009 R
/R
/$ 00000000
/O SYM                  List symbol table
/E                      End of patches
```

Notes:

1. No print, no halt on error, get module definitions, save patches.
2. Branch to the patch area from location `100` of `H.REXS`.
3. CI R6, 1 inserted in the next free location of patch area.
4. BCF EQ,`5E5` of `H.REXS` inserted.

9.7 Automatic Mounting of Public Volumes

The last function of J.INIT before exiting is the mounting of any volumes in the M.MOUNT file.

See Chapter 10 of this volume for more information.

10 System Administrator Services

10.1 Introduction

The system administrator (SA) implements and controls the system and resource protection facilities provided by MPX-32. System protection is concerned primarily with user access to the system. The system protection facilities are implemented by the M.KEY and M.PRJCT file.

While not a direct function of the M.KEY and M.PRJCT files, resource protection facilities are not fully utilized unless these files are present and set up properly.

An individual can be designated to have the responsibilities of initializing terminals, responding to system console messages as necessary for system operation, and initializing floppy disk media.

Within a task, a privileged function can be performed for an unprivileged user with an SVC call. At the command processor level, this is done by allowing users to execute tasks that have the SA attribute. This attribute is established when the load module or executable image is created.

The SA attribute permits the task to:

- access resources without authorization checking
- use the M.CALL macros (SVC 0)
- mount public volumes
- update the current system date and time
- invoke the dual-processor shared-volume recovery task (J.UNLOCK)
- set and override timeout and logon counters for security purposes

The SA attribute is honored only if the load module or executable image file resides on the system volume and the owner of the file is SYSTEM.

Any task executed with OWNER=SYSTEM also has the SA attribute.

A task cataloged with SA can also specify the NODEBUG attribute to prevent unauthorized users from modifying the code.

M.CALL macros can be used by SA tasks that are not privileged. Therefore, programs for such tasks should be written only by users with a thorough knowledge of the operating system.

The operating system also provides various optional services to control access to the system, accounting services, and user diagnostics. These services are invoked through the following system files:

- M.KEY — Defines the set of authorized users of the system. Absence of this file prevents control of access to the system. This file is managed by the KEY program.
- M.PRJCT — Defines the set of authorized project group names for resource access and accounting. This file is managed by J.PRJCT.
- M.ACCNT — Records accounting information for terminal sessions and jobs
- M.CNTRL — Provides automatic command file processing at logon
- M.ERR — Contains system error explanations. This file can be modified to include user error messages.
- M.MOUNT — Provides automatic mounting of specified volumes during IPL

If the M.KEY, M.PRJCT, M.ACCNT, or M.ERR file is recreated, the new file is immediately effective. If control switch 4 is set at IPL, these four files are ignored while the system is running.

10.2 M.KEY File

The M.KEY file controls access to the system.

Any user may logon to the system if the M.KEY file does not exist. All users who logon while the M.KEY file does not exist will be assigned the working directory @SYSTEM(SYSTEM).

Once the M.KEY file has been created, only users established in this file may use the system.

For each user in the file, the SA can specify the following attributes or restrictions:

- password — a 1- to 16-character owner password required for logon
- key — a 1- to 8-character owner key required for logon
- default project group name
- default volume name
- default working directory
- default tab settings
- flagwords — a series of directives that determine access to various parts of the system

If a user has been assigned a default volume or default working directory that cannot be located when the user logs on to the system, the following message is displayed:

```
*YOUR DIRECTORY IS UNAVAILABLE, PLEASE CHANGE DIRECTORY
```

When this occurs, the user may only execute TSM directives. The user's volume and directory name will be shown as asterisks by the $SHOW directive.

To prevent access to the M.KEY file by any user except SYSTEM, start up the system with control switch 4 set and create an M.KEY file under owner name SYSTEM using the following VOLMGR directive:

```
VOL>CREATE @SYSTEM(SYSTEM)M.KEY ACCE=OW(R W A D)-
VOL>ACCE=PR() ACCE=OT() ZERO=Y AUTO=N SIZE=N -
VOL>REPLACE=Y
```

Restart the system with control switch 4 reset.

When these access rights are specified, users cannot change their password or key at logon and the PASSWORD and KEYWORD tasks do not function.

10.2.1 Using KEY

The system administrator attribute is required to run the KEY program. The primary file required to use KEY is an input file containing M.KEY information for each owner.

M.KEY is supplied on the Master SDT with owner name SYSTEM.

If the M.KEY file is recreated, the new file is immediately effective. If control switch 4 is set at IPL, the M.KEY file is ignored while the system is running.

The input file is prepared using the Edit STORE command. This file must be assigned to logical file code INP. Output is unblocked and automatically assigned to the system file M.KEY.

When recreating the M.KEY file, valid owner names must be specified before logging off the system so that valid names exist for future logons.

```
TSM>$ASSIGN INP TO FILENAME
TSM>$KEY
```

10.2.2 Directives

KEY directives have one or more associated parameters. Some parameters can be preceded by a keyword and equal sign. Multiple parameters can be separated by TSM delimiters: spaces, commas, equal sign, and parentheses.

Continuation of the directive to the next physical line is specified by a comma following the last parameter on the line.

KEY directives are summarized below and described individually on the following pages.

Directive	Description
ADD	authorizes a new user access to the system
CHANGE	changes the attributes of an existing owner
DEFAULTS	resets system defaults and establishes defaults for subsequent ADD commands
DELETE	removes an existing owner from the M.KEY file
LOG	lists existing owners and all attributes except their key
NEWFILE	clears the M.KEY file of all existing entries
X	indicates end of file and exits

10.2.2.1 ADD Directive

The ADD directive authorizes new users access to the system and specifies any restrictions applicable to them.

Syntax

ADD [OWNER=]*name* [*attribute*]

[OWNER=]*name*
name is the 1- to 8-character owner name

[*attribute*] is one of the following attributes or restrictions to be in effect for the specified owner:

PASSWORD=*name*
name is the 1- to 16-character owner password associated with this owner. If this parameter is not specified, the default is no password.

KEY=*name*
name is the 1- to 8-character key associated with this owner. If not specified, the default is no key. For a list of restricted keywords refer to Table 10-1.

VOLUME=*name*
name is the 1- to 16-character default volume name associated with this owner. If not specified, the default is SYSTEM.

DIRECTORY=*name*
name is the 1- to 16-character default directory name associated with this owner. If not specified, the default is SYSTEM.

PROJECT=*name*
name is the 1- to 8-character default project group name associated with this owner. If not specified, the default is SYSTEM.

TABS=*tab* [,*tab*] ...

sets decimal tab positions to be associated with this owner. As many as 8 tab positions can be set. If not specified, default tabs are set at 10, 20, and 36.

[**FLA**GS=]*restrict*

restrict is one of the following:

ALL specifies all restrictions apply to this owner

NONE specifies no restrictions apply to this owner

ALLEXCEPT *flist*
specifies all restrictions except the following listed key words (*flist*) apply to this owner

flist specifies the key words whose operations cannot be performed by this owner. See Table 10-1.

Table 10-1
Restricted Keywords

The following keywords disable the corresponding OPCOM directives:

ABORT	DISABLE	MODE	REQUEST
ACTIVATE	DISCONNECT	MODIFY	RESUME
BATCH	DISMOUNT	MOUNT	SEARCH
BREAK	DUMP	OFFLINE	SEND
CONNECT	ENABLE	ONLINE	SETTIMER
CONTINUE	ESTABLISH	PURGEAC	SNAP
DELETETIMER	HOLD	REDIRECT	STATUS
DEPRINT	KILL	REPRINT	SYSASSIGN
DEPUNCH	LIST	REPUNCH	TIME

When the following keywords are specified, the corresponding restrictions apply:

Keyword	Restriction
CATPRIV	Cannot catalog privileged programs.
CHANGEDEF	Cannot change working directory or project group.
MDT	Cannot use the J.MDTI utility*
OWNERACC	Cannot access tasks with different owner name
PRIORITY	Cannot use the TSM $URGENT directive
PRIVILEGE	Cannot run privileged programs other than OPCOM
REMOVE	Cannot use TSM $REMOVE directive
RESTART	Cannot use the TSM $RESTART directive
USERFLAGS (*n*)	Decimal number in range 0 to 255 for use by user for security. User programs may examine this number in bits 56 to 63 of T.ACCESS.

When the following keywords are specified, the corresponding defaults apply:

Keyword	Default
NOCOMMAND	OPTION NOCOMMAND in TSM
SEQUENTIAL	If more than one job is submitted by a TSM $SUBMIT directive, the jobs execute sequentially. If sequential execution is required for jobs run using a BATCH or RUN directive, S must be specified on the $JOB directive line.

* The restrictions on this keyboard apply to only the KEY ADD directive.

10.2.2.2 CHANGE Directive

The CHANGE directive changes the attributes of an existing owner of the system.

Syntax

CHANGE [**OWN**ER=]*name* [*attribute*]

[**OWN**ER=]*name*
name is the 1- to 8-character owner name

[*attribute*] is one of the following attributes or restrictions to be in effect for the specified owner:

PASSWORD=*name*
name is the 1- to 16-character owner password associated with this owner. If this parameter is not specified, the default is no password.

KEY=*name*
name is the 1- to 8-character key associated with this owner. If not specified, the default is no key. For a list of restricted keywords refer to Table 10-1.

VOLUME=*name*
name is the 1- to 16-character default volume name associated with this owner. If not specified, the default is SYSTEM.

DIRECTORY=*name*
name is the 1- to 16-character default directory name associated with this owner. If not specified, the default is SYSTEM.

PROJECT=*name*
name is the 1- to 8-character default project group name associated with this owner. If not specified, the default is SYSTEM.

TABS=*tab* [,*tab*] ...
sets decimal tab positions to be associated with this owner. As many as 8 tab positions can be set. If not specified, default tabs are set at 10, 20, and 36.

[**FLA**GS=]*restrict*
restrict is one of the following:

ALL specifies all restrictions apply to this owner

NONE specifies no restrictions apply to this owner

ALLEXCEPT *flist*
specifies all restrictions except the following listed key words (*flist*) apply to this owner

flist adds the listed keywords to the existing restrictions for this owner. See Table 10-1.

10.2.2.3 DEFAULTS Directive

The DEFAULTS directive establishes defaults for subsequent ADD directives.

Each DEFAULTS remains in effect until the next DEFAULTS is specified. When the next DEFAULTS is specified, the default for any keyword not specified is the SYSTEM default. When more than one keyword default is to be defined for a group of users, all defaults must be specified with one DEFAULTS directive as shown in the example of directive usage.

Syntax

DEFAULTS [*attribute*]

[*attribute*] are the following attributes or restrictions to be used as defaults for subsequent ADD directives:

KEY=*name* *name* is the 1- to 8-character key associated with the specified owner. If not specified, the default is no key. For a list of restricted keywords refer to Table 10-1.

VOLUME=*name*
name is the 1- to 16-character default volume name associated with the specified owner. If not specified, the default is SYSTEM.

DIRECTORY=*name*
name is the 1- to 16-character default directory name associated with the specified owner. If not specified, the default is SYSTEM.

PROJECT=*name*
name is the 1- to 8-character default project group name associated with the specified owner. If not specified, the default is SYSTEM.

TABS=*tab* [,*tab*] ...
sets decimal tab positions to be associated with the specified owner. As many as 8 tab positions can be set. If not specified, default tabs are set at 10, 20, 36, and 64.

[FLAGS=]*restrict*

restrict is one of the following:

ALL specifies all restrictions apply to the specified owner

NONE specifies no restrictions apply to the specified owner

ALLEXCEPT *flist*
specifies all restrictions except the following listed key words (*flist*) apply to the specified owner

flist specifies the key words whose operations cannot be performed by the specified owner. See Table 10-1.

10.2.2.4 DELETE Directive

The DELETE directive removes an existing owner from the M.KEY file.

Syntax

DELETE [**OWN**ER=]*name*

[**OWN**ER=]*name*
name is the 1- to 8-character owner name

10.2.2.5 LOG Directive

The LOG directive lists existing owners and all attributes except their key.

Syntax

LOG [[**OWN**ER=]*name*]

[[**OWN**ER=]*name*]
name is the 1- to 8-character owner name whose attributes are to be displayed. If not specified, all owners attributes are displayed.

10.2.2.6 NEWFILE Directive

The NEWFILE directive clears the M.KEY file of all existing entries. If present, it must be the first directive specified.

Syntax

NEWFILE

10.2.2.7 X Directive

The X directive indicates end-of-file and exit.

Syntax

X

10.2.3 Examples of Directive Usage

```
NEWFILE
ADD OWNER=SYSTEM VOLUME=SYSTEM DIRECTORY=SYSTEM
DEFAULTS PROJECT=WORK TABS=7,12,24 FLAGS=NONE
ADD OWNER=OWN1 VOLUME=VOL1 DIRECTORY=DIR1
ADD OWNER=OWN2 VOLUME=VOL2 DIRECTORY=DIR2
ADD OWNER=OWN3
DEFAULTS VOLUME=TEST PROJECT=EXAM
ADD OWNER=OWN4 KEY=0416 TABS=10,25,50
CHANGE OWNER=OWN3 VOLUME=VOL3
ADD OWNER=OWN5 FLAGS=ALLEXCEPT MOUNT,SEARCH,PRIVILEGE,SEQUENTIAL
ADD OWNER=OWN6
DEFAULTS VOLUME=TEST DIRECTORY=DIR3 PROJECT=UTIL FLAGS=ABORT,HOLD,
MOUNT,MODIFY,PRIVILEGE,PRIORITY,SEQUENTIAL TABS=9,18,36
ADD OWNER=OWN7 KEY=1205
ADD OWNER=OWN8 KEY=1103
ADD OWNER=OWN9 KEY=0613
DELETE OWN3
LOG
X
```

10.3 M.PRJCT File

The M.PRJCT file contains the project group names that are valid to use with the TSM CHANGE PROJECT directive and are used by the accounting utility M.ACCNT. If a M.PRJCT file does not exist, any project group name is valid. If an M.PRJCT file does exist and an owner has a default project name not contained in M.PRJCT, no project name is extablished at logon.

When using the accounting utility, project group names can be established for each owner name in two ways:

1. Using the Volume Manager, an M.PRJCT file can be created under the owner SYSTEM as a nonextendible file that is not accessible by PROJECTGROUP or OTHER. The file should be zeroed. Eight entries per block can be placed in this file from an editor-created user file using the PROJECT program J.PRJCT. The output from the PROJECT program need not be assigned; output automatically goes to M.PRJCT unblocked. Only users with the system administrator attribute can use the PROJECT program.

   Example

   ```
   ENTER YOUR OWNERNAME:SYSTEM
   TSM>VOLMGR
   VOL>CREATE @SYSTEM(SYSTEM)M.PRJCT AUTO=N ZERO=Y-
   VOL>ACCESS=PROJECTGROUP()-
   VOL>ACCESS=OTHER() SIZE=4
   VOL>X
   TSM>$ASSIGN INP TO PFILE
   TSM>PROJECT
   ```

2. Using the KEY utility, default project group names can be established for each owner name.

Project group names can be changed by a user with the TSM $CHANGE PROJECT directive. When a project group name is changed, the account utility is terminated for the previous group name and initiated for the new group name. The change remains in effect until changed again or the user logs off the system. When the user logs on the system, default project group names are re-established.

10.3.1 Using the PROJECT Program

The primary file required to use PROJECT is an input file containing M.PRJCT information for each owner.

This input file (*filename* in the following example) is prepared using the Edit STORE directive. It must be assigned to the logical file code INP. Output is unblocked and automatically assigned to the system file M.PRJCT.

Example

```
TSM>$ASSIGN INP TO filename
TSM>$PROJECT
```

Only owners with the system administrator attribute can run the PROJECT program.

If the M.PRJCT file is modified or recreated, the changed or new file does not become effective until the system is rebooted. If control switch 4 is set at IPL, the M.PRJCT file is ignored while the system is running.

10.3.2 Directives

Input to the PROJECT program is a series of directives that are summarized below and described in detail on the following pages.

PROJECT directives have one or more associated parameters. Some parameters can be preceded by a keyword and equal sign. Multiple parameters can be separated by the normal TSM delimiters: spaces, commas, equal sign, and parentheses.

Continuation of a directive to the next physical line is specified by a comma following the last parameter on the line.

PROJECT directives are summarized below and described individually on the following pages.

Directive	Description
ADD	authorizes a new project group name
CHANGE	changes a key associated with a project group name
DELETE	removes a project group name from the PROJECT file
LOG	lists existing project group names
NEWFILE	clears the M.PRJCT file of all existing entries
X	indicates end-of-file and exits

10.3.2.1 ADD Directive

The ADD directive adds new project group names.

Syntax

ADD [**PRO**JECT=]*pname* [**KEY**=*key*]

[**PRO**JECT=]*pname*
: *pname* is the 1- to 8-character project group name to be added

[**KEY**=*key*]
: *key* is the 1- to 8-character key to be associated with this project group name

10.3.2.2 CHANGE Directive

The CHANGE directive changes an existing key associated with a project group name or establishes a key with an existing project group name that does not have an associated key. A key cannot be deleted with this directive. If a key is to be deleted, the DELETE directive must be used to delete the project name; the ADD directive must then be used to re-establish the project name without a key.

Syntax

CHANGE [**PRO**JECT=]*pname* [**KEY**=*key*]

[**PRO**JECT=]*pname*
: *pname* is the 1- to 8-character project group name

[**KEY**=*key*]
: *key* is the new 1- to 8-character key to be associated with this project group name

10.3.2.3 DELETE Directive

The DELETE directive removes an existing project group name from the PROJECT file.

Syntax

DELETE [**PRO**JECT=]*pname*

[**PRO**JECT=]*pname*
: *pname* is the 1- to 8-character project group name to be deleted

10.3.2.4 LOG Directive

The LOG directive lists existing project group names.

Syntax

LOG [**PRO**JECT=]*pname*

[**PRO**JECT=]*pname*

pname is the 1- to 8-character project group name whose authorization is to be listed. If not specified, all project group names are displayed.

10.3.2.5 NEWFILE Directive

The NEWFILE directive clears the M.PRJCT file of all existing entries. If present, it must be the first directive specified.

Syntax

NEWFILE

10.3.2.6 X Directive

The X directive indicates end-of-file and exit.

Syntax

X

10.3.3 Examples of Directive Usage

```
NEWFILE
ADD PROJECT=SYSTEM
ADD PROJECT=WORK
ADD PROJECT=EXAM
ADD PROJECT=UTIL
ADD TEST
ADD LANG KEY=XYZ
ADD DOC KEY=JM
CHANGE PROJECT=LANG KEY=DR
CHANGE TEST KEY=CASE
DELETE PROJECT=DOC
ADD DOC
LOG
X
```

10.4 M.ACCNT File

The job accounting program indicates elapsed time, CPU time, and IPU time for all jobs. The M.ACCNT file must be created by the Volume Manager under the owner name SYSTEM as a nonextendible zeroed file with read access by PROJECTGROUP and OTHER. Its size is determined by the need of the individual site. Its format is 16 word entries in an unblocked 192W physical record. This file is where job accounting information is collected for use by TSM.

Example

```
ENTER YOUR OWNERNAME:SYSTEM
TSM>VOLMGR
VOL>CREATE @SYSTEM(SYSTEM)M.ACCNT AUTO=N ZERO=Y SIZE=100-
VOL>ACCESS=PROJECTGROUP(R)-
VOL>ACCESS=OTHER(R)
```

The M.ACCNT file is created as a nonextendible file on volume SYSTEM and directory SYSTEM. PROJECTGROUP and OTHER have read access. As many as 1200 entries can be specified.

If the M.ACCNT file is recreated, the new file does not become effective until the system is rebooted. If control switch 4 is set at IPL, the M.ACCNT file is ignored while the system is running.

Default project group names for accounting purposes can be established with the M.KEY utility.

Data collected by the accounting program can be retrieved with the OPCOM LIST command. Refer to the OPCOM chapter in Reference Manual Volume II.

The format of M.ACCNT file entries follows.

M.ACCNT File

Word	0 – 7 8 – 15 16 – 23 24 – 31
Word 0-1	Owner name (1- to 8-character ASCII). See Note 1.
2-3	Project (1- to 8-character ASCII). See Note 2.
4-5	Date (*mm/dd/yy*) (ASCII). See Note 3.
6-7	Logon time (*hh/mm/ss*) (ASCII). See Note 4.
8-9	Elapsed time (*hh:mm:ss*) (ASCII). See Note 5.
10	Raw CPU time (Binary). See Note 6.
11	Raw IPU time (Binary). See Note 7.
12-13	Origin (1- to 8-character ASCII). See Note 8.
14-15	Reserved

Notes:

1. Owner name – the 1- to 8-character ASCII owner name associated with the job
2. Project – the 1- to 8-character ASCII alphanumeric project name or number associated with the job
3. Date – the ASCII numeric date associated with the job
4. Logon time – the ASCII numeric time of day on the 24-hour clock the user signed on the system
5. Elapsed time – the total time (24-hour clock) the user was signed on the system in ASCII
6. Raw CPU time – the actual number of 38.4 microsecond intervals of CPU time for the job (unformatted equivalent of CPU time) in binary
7. Raw IPU time – the actual number of 38.4 microsecond intervals of IPU time for the job (unformatted equivalent of IPU time) in binary
8. Origin – the ASCII task pseudonym for the accounting session

10.5 M.CNTRL File

The M.CNTRL file is a TSM command file selected by J.TSM automatically when a user logs on. If the M.CNTRL file exists, it must be located in the system directory.

As the M.CNTRL file can contain TSM directives and comments, it can establish defaults, send messages, and further restrict access to the TSM environment.

Example

```
EDT>COL
1.   NOTE LOG ON AT
2.   !TIME
3.   <cr>
EDT>STO M.CNTRL SYS
```

This causes the time and date to be automatically displayed when a user logs on the system. NOTE indicates the line is a comment line.

Notes:

Attempts to break out of the M.CNTRL file are ignored.

Errors in M.CNTRL file processing result in normal error processing for control files.

10.6 M.ERR File and *xx*.ERR Files

The M.ERR file contains system abort codes and messages, and should not be modified. The *xx*.ERR files contain messages for unbundled products and user abort codes. For example, FT.ERR contains FORTRAN 77+ abort messages.

The abort code format is:

xxnn

x	is an alphabetic character
n	is a numeral

The file SJ.XX.ER is provided on the SDT and can be modified with other abort messages by using the Text Editor (EDIT). When an abort code cannot be found in the M.ERR file, J.TSM appends .ERR to the abort code's prefix. Using the resulting name, J.TSM attempts to allocate a file in the system directory. If allocation occurs, the file is searched for the error code.

10.6.1 Creating *xx*.ERR File

Modify SJ.XX.ERR as follows:

1. Enter EDIT.
2. Use file SJ.XX.ER.
3. Modify the prefix at label `STARTX`.

   Example:

   ```
   STARTX PREFIX C'MD' MEDIA ERRORS
   ```
4. Remove the example messages.
5. Add messages using `C` strings.
6. After each message, call macro MSG *xx*,*nn* where *xx* is the prefix specified at `STARTX`; *nn* is the abort number in ascending order.

   Example (starts in the second column):

   ```
   DATA C'ERROR ENCOUNTERED READING SYC FILE'
   MSG LM,01
   DATA C'ERROR ENCOUNTERED WRITING TO SLO'
   MSG LM,03
   ```
7. Save the completed file.

   Example:

   ```
   EDT>SAVE @SYSTEM(SOURCE)SJ.xx.ER
   ```
   (*xx* is the prefix specified at `STARTX`)

8. Exit EDIT.
9. Submit the following job stream to install the new error file:

`$JOB NEWERR OWNER`	
`$ASSIGN SI TO SJ.xx.ER BLOCK=Y`	(SJ.*xx*.ER is the file previously saved in Step 7 by the Text Editor (EDIT))
`$ASSIGN BO TO OJ.xx.ER BLOCK=Y`	(OJ.*xx*.ER must be created prior to this assignment)
`$OPTION 1 4`	
`$ASSEMBLE`	
`$AS SGO TO OJ.xx.ER BLOCK=Y`	
`$IFF PATH=@SYSTEM^(SYSTEM)xx.ERR GO`	(*xx* is the prefix entered in Step 3. This text ensures that an existing abort file code is not destroyed).

```
$NOTE ERROR - ATTEMPT MADE TO WRITE OVER EXISTING ABORT FILE
$GOTO END
$DEFNAME GO
$CATALOG
CATALOG xx.ERR                          (xx is the prefix entered in Step 3)
$DEFNAME END
$EOJ
$$
```

Abort messages are immediately in effect without rebooting the system.

10.7 Terminal/ALIM/ACM Initialization (INIT)

Terminal/ALIM/ACM initialization is the process of defining hardware characteristics of TSM and non-TSM devices. For purposes of this section, TSM devices are defined as terminals, device type code X'0C' TY, connected through model 9110 ALIM or 8510, 8511, or 8512 Eight-line Asynchronous Communications controllers. Non-TSM devices are defined as devices other than terminals; the device type code is not TY, but the device can be connected through these same controllers.

Hardware characteristics are typically defined by a user-created system file named LOGONFLE, or in its absence, by a set of system supplied defaults. Characteristics include baud rate, parity, and half or full duplex. The wakeup (ring) character for all terminals is also defined as a hardware characteristic.

Initialization of a TSM device is handled automatically when a system is installed or restarted, regardless of the existence of a LOGONFLE. Initialization of a non-TSM device requires a LOGONFLE entry defining the characteristics of the non-TSM device and the keyword NOTSM.

Although the IOP console is normally treated as a TSM device, it is not initialized by INIT.

10.7.1 The LOGONFLE

The LOGONFLE must contain the logon records in blocked, uncompressed format. The Editor STORE directive can create this file.

The form of a LOGONFLE is shown below. LOGONFLE must contain a record for the wakeup character definition and one record for each device definition. Only characters 1 to 72 of each record are interpreted. An asterisk (*) in column 1, from Record 2 on, indicates the line is a comment line. A semicolon (;) or an exclamation point (!) in any position of a line, from Record 2 on, permits comments to follow on the line.

Record 1: *wakeup*
Record 2: *ccaa field field ...*
.
.
.
EOF

wakeup is a 2-digit hexadecimal number defining the character which must be typed at interactive terminals to start a logon sequence.

ccaa is the channel number and subaddress of the device to initialize. Must be supplied in LOGONFLE for each device. If LOGONFLE was not created, the system default is to initialize all addresses defined at SYSGEN as device type TY with the system default parameters. If channel *cc* exists, but no device is connected to subaddress *aa*, J.TINIT will mark the device online but note them as failed (dead) devices; if *cc* does not exist, J.TINIT will mark the device offline and dead.

field is a 1- to 8-character keyword, (the first 4 characters are significant,) describing the characteristics of a device. Fields can be entered in any order and can be duplicated. Fields are evaluated left to right; later entries overrule earlier entries. See Table 10-2 for valid keywords and their significance to ALIM and 8-Line Async devices. If a LOGONFLE is not created, all devices defined as device type TY at SYSGEN are initialized with the system default parameters (see section 2.2 of this chapter). Before LOGONFLE is created, J.INIT or J.TINIT will mark all TY devices online, but note them as failed devices (dead terminals). TSM devices not included in the LOGONFLE once it is created retain the system default parameters and remain online and dead.

Table 10-2
LOGONFLE Field Keywords

Category	Keyword	ALIM	8-Line Async
Baud Rate	19200	not used	19200 bps
	9600	9600 bps	9600 bps
	7200	7200 bps	7200 bps
	4800	4800 bps	4800 bps
	3600	3600 bps	3600 bps
	2400	2400 bps	2400 bps
	2000	not used	2000 bps
	1800	1800 bps	1800 bps
	1200	1200 bps	1200 bps
	900	900 bps	not used
	600	600 bps	600 bps
	300	300 bps	300 bps
	150	150 bps	150 bps
	134	134.5 bps	134.5 bps
	110	110 bps	110 bps
	75	75 bps	75 bps
	50	50 bps	50 bps
	EXT	external rate	not used
Parity	ODD	odd parity	odd parity
	EVEN	even parity	even parity
	NONE	no parity	no parity
Duplex	HALF	half duplex	half duplex
	FULL	full duplex	full duplex
	full duplex bit set in UDT		
Stop Bits	S1	1 stop bit	1 stop bit
	S1.5	1.5 stop bits	1.5 stop bits
	S2	2 stop bits	2 stop bits
Character Size	8	8-bit characters	8-bit characters
	7	7-bit characters	7-bit characters
	6	6-bit characters	6-bit characters
	5	5-bit characters	5-bit characters
Miscellaneous	GRAP	graphic device	graphic device
	MODEM	sets modem bit in UDT	sets modem bit in UDT
	REMOTE	sets dial-up bit in UDT	sets dial-up bit in UDT
	NOTSM	not a TSM device	not a TSM device
	INIT	initialization data present	initialization data present
	LGC=*nnn*	max number of logon attempts allowed	max number of logon attempts allowed
	LGT=*sss*	max time allowed at logon prompt in seconds	max time allowed at logon prompt in seconds
	TMO=*mmm*	max wait time for a terminal read in minutes	max wait time for a terminal read in minutes

Note: System-wide default values can be set for LGC, LGT, and TMO by specifying LGC, LGT, and TMO in a comment line that occurs before the first terminal definition. Refer to the Sample LOGONFLE section in this chapter.

10.7.2 Security Counters

MPX-32 provides the system administrator with controls that protect the system from unauthorized access to maintain system integrity. These controls are counters, specified in the LOGONFLE, that limit access as follows:

- the number of logon attempts allowed
- the time spent when logging on
- the amount of time that a terminal can remain I/O inactive

The following section discusses these counters and provides an example of their usage.

10.7.2.1 Logon Attempt Counter

The logon attempt counter limits the number of logon attempts. This counter can vary from terminal to terminal. The system administrator specifies the number of logon attempts in the LOGONFLE as follows:

`LGC=`*nnn*

nnn is the number of logon attempts, 0 to 255, allowed before the terminal is marked offline. If *nnn* is 0, there is no limit on the number of logon attempts. The default is 0.

For more information about the LOGONFLE, refer to section 10.8.1 in this chapter.

To change the logon attempt counter, the system administrator must change the LOGONFLE. To override the counter until the next terminal initialization, enter the following:

`TSM>LOGCNT` *termaddr nnn*

termaddr specifies the address of the terminal. If *termaddr* is not specified, LOGCNT uses the value established as the system default in the LOGONFLE.

nnn specifies the number of logon attempts, 0 to 255, allowed. The default is 0.

If a terminal is marked offline, the system administrator can enable the terminal with the TSM $ENABLE command. See Chapter 1 of the MPX-32 Reference Manual Volume II for more information.

10.7.2.2 Logon Timeout Counter

The logon timeout counter limits the time that can be spent at a logon prompt. If this counter expires, the terminal is marked offline and the logon attempt counter is incremented. The system administrator specifies the logon timeout counter in the LOGONFLE as follows:

`LGT=`*sss*

sss specifies the number of seconds, 0 to 255, that a terminal waits at a logon prompt before being logged off. The default is 0.

To change the logon timeout counter, the system administrator must change the LOGONFLE. To override the counter until the next terminal initialization, enter the following:

`TSM>LOGTIME` *termaddr sss*

termaddr specifies the address of the terminal. If *termaddr* is not specified, LOGTIME uses the value established as the system default in the LOGONFLE.

sss specifies the number of seconds, 0 to 255, that the terminal waits at a logon prompt. The default is 0.

10.7.2.3 Terminal Timeout Counter

The terminal timeout counter limits the time that a terminal can remain I/O inactive. The system administrator specifies the terminal timeout counter in the LOGONFLE as follows:

`TMO=`*mmm*

mmm specifies the number of minutes, 0 to 999, that a terminal can be I/O inactive before it is logged off. If *mmm* is 0, there is no limit. The default is 0.

To change the terminal timeout counter, the system administrator must change the LOGONFLE. To override the counter until the next terminal initialization, enter the following:

`TSM>TERMOUT` *termaddr mmm*

termaddr specifies the address of the terminal. If *termaddr* is not specified, TERMOUT uses the value established as the system default in the LOGONFLE.

mmm specifies the number of minutes, 0 to 999, that a terminal can be I/O inactive before it is logged off. If *mmm* is 0, there is no limit. The default is 0.

If a terminal is marked offline, the system administrator can enable the terminal with the TSM $ENABLE command. See Chapter 1 of the MPX-32 Reference Manual Volume II for more information.

10.7.2.4 Security Counters Example

The following is an example that sets the logon attempt counter, the logon timeout counter, and the terminal timeout counter.

```
05        !RING IN CTRL/E)
*      LGC=5 LGT=45 TMO=120 !system defaults

7EA0  19200 FULL EVEN 7 S1                  !no changes
7EA1  9600 FULL EVEN S1 EXON                !no changes
7EA2  19200 FULL EVEN 7 S1WXON TMO=60       !Sets Term Timeout Ctr
7EA3  9600 FULL EVEN 7 S1 WXON              !no changes
7EA4  19200 FULL EVEN 7 S1 LGT=120          !sets logon timer
7EA5  19200 FULL EVEN 7 S1                  !no changes
7EA6  19200 FULL EVEN 7 S1 WXON             !no changes
7EA7  1200 FULL NONE 6 WXON REMOTE LGC=3 !sets logon counter
```

10.7.3 ALIM Terminal Record Syntax and Defaults

Syntax

ccaa [*baud*] [**HALF|FULL**] [**ECHO|NOECHO**] [*parity*] [*charsize*] [*stopbits*] [**GRAP**] [**MODEM**] [**REMOTE**] [**NOTSM**] [**INIT** *value*] [**LGC=***nnn*] [**LGT=***sss*] [**TMO=***mmm*]

ccaa is the channel and subaddress of the device.

[*baud*] is the baud rate: 9600, 7200, 4800, 3600, 2400, 1800, 1200, 900, 600, 300, 150, 134, 110, 75, 50, or EXT. If EXT is entered, the baud rate is set externally. The default is 9600 baud.

[**HALF|FULL**]
is the half or full duplex operation. The default is HALF.

[**ECHO|NOECHO**]
specifies characters are or are not to be echoed by the computer as they are received. The default is ECHO for FULL duplex or NOECHO for HALF duplex.

[*parity*] is ODD, EVEN, or NONE. The default is EVEN. If NONE is specified, the serial character is smaller due to the absence of the parity bit.

[*charsize*] is the character size: 5, 6, 7, or 8. The default is 7.

[*stopbits*] is the number of stop bits: S1, S1.5, or S2. The default is S1.

[**GRAP**] sets the graphic bit in the UDT. GRAP specifies that standard message processing should be suppressed for this device. The default is not graphic device.

[**MODEM**] sets modem bit in UDT. The default is not set.

[**REMOTE**] sets dial-up bit in UDT. The default is not set. Also, sets switched mode. The default is private mode.

[**NOTSM**] specifies a non-TSM device. The default is TSM device.

[**INIT** *value*]
indicates the presence of hexadecimal initialization data. Five words of hexadecimal value in the ALIM initialization format. See section 10.7.3.1, ALIM Initialization Format.

[**LGC=***nnn*]
nnn specifies the number of logon attempts, 0 to 255, allowed before the terminal is marked offline. If *nnn* is 0, there is no limit on the number of logon attempts. The default is 0.

[**LGT=***sss*] *sss* specifies the number of seconds, 0 to 255, that a terminal waits at a logon prompt before being logged off. The default is 0.

[**TMO=***mmm*]
mmm specifies the number of minutes, 0 to 999, that a terminal can remain I/O inactive.

The following table shows possible ALIM terminal configurations and their resulting operations.

SYSGEN Device Directive, FULL Option Specified	LOGONFLE FULL/HALF Option Specified	LOGONFLE ECHO/NOECHO Option Specified	Resulting Terminal Operation
yes	full	echo	half,echo
yes	full	noecho	full,noecho
yes	half	echo	half,echo
yes	half	noecho	half,noecho
no	full	echo	half,echo
no	full	noecho	half,noecho
no	half	echo	half,echo
no	half	noecho	half,noecho

10.7.3.1 ALIM Initialization Format

Fields containing an asterisk (*) are required; the remaining fields are optional. If not specified, the default is OFF.

Word	0–7	8–15	16–23	24–31
1	Channel time-out value*. See Note 1.			
2	Mode *. See Note 2.	Baud *. See Note 3.	Format. See Note 4.	Zero
3	Code. See Note 5.	STX. See Note 6.	EXT. See Note 7.	CHAR A. See Note 8.
4	CHAR B. See Note 9.	CHAR C. See Note 10.	CHAR D (ring). See Note 11.	S1. See Note 12.
5	S2. See Note 13.	Zero		

Notes:

1. Bits 0-31 specify the channel time-out value in seconds. For example, X'0000005A' gives a time-out value of 90 seconds.

2. Mode bits are defined as follows:

Bits	Description
0-1	Stop bits (SB) are interpreted as follows:

Bit 0	Bit 1	SB
0	0	invalid
0	1	1 stop bit
1	0	1.5 stop bits
1	1	2 stop bits

2 Parity selection (PS) is interpreted as follows:

Bit Setting	Description
0	odd parity
1	even parity

3 Parity enable (PE) is interpreted as follows:

Bit Setting	Description
0	disable parity
1	enable parity

4-5 Character length (CL) is interpreted as follows:

Bit 4	Bit 5	CL
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

6-7 Baud rate factor (BRF) is interpreted as follows:

Bit 6	Bit 7	BRF
0	0	synch mode
0	1	1X
1	0	16X
1	1	64X

3. Baud rate bits are defined as follows:

Bits	Description
8-11	Zero
12-15	With 16X selected in bits 6-7, baud rate is interpreted as follows:

Bit 12	Bit 13	Bit 14	Bit 15	Baud Rate
0	0	0	1	50
0	0	1	0	75
0	0	1	1	110
0	1	0	0	134.5
0	1	0	1	150
0	1	1	0	300
0	1	1	1	600
1	0	0	0	900
1	0	0	1	1200
1	0	1	0	1800
1	0	1	1	2400
1	1	0	0	3600
1	1	0	1	4800
1	1	1	0	7200
1	1	1	1	9600
0	0	0	0	EXT

4. Format bits are defined as follows:

Bits	Description
16	enable hardware ring detection
17	enable break detection
18	inhibit 1ms RTS off delay
19	full or half duplex interpreted as follows:

Bit Setting	Description
0	half duplex (HDX)
1	full duplex (FDX)

Bits	Description
20	Switch indicator is interpreted as follows:

Bit Setting	Description
0	private
1	switched line

Bits	Description
21	Must be set for types 103 and 212 modems

Bit Setting	Description
0	switched
1	constant

Bits	Description
22-23	Zero

5. Code bits are defined as follows:

Bits	Description
0-3	zero
4-7	Code bits (4-bit hexadecimal value) are interpreted as follows:

Value	Description
0	no action
1	terminate on CHAR A
2	terminate on CHAR A or B
3	terminate on CHAR A, B, or C
4	terminate on CHAR A* followed by B*
5	terminate on CHAR A* followed by B* followed by C*

* These characters must be different from STX, ETX, ring, or strip characters

6. Start of text character (STX) bits are defined as follows:

Bits	Description
8	Action bit interpreted as follows:

Bit Setting	Description
0	recognize character
1	inhibit character recognition

Bits	Description
9-15	Start of text character (STX) to be recognized (7-bit ASCII character). The character must be different from ETX, strip, ring, and terminate characters.

Upon recognition, unblind mode is set causing this character and subsequent characters to be passed to memory. Blind mode, which deletes characters or blocks of characters, is set by either a receive blind order or the recognition of an ETX character during a receive blind order.

7. End of text character (ETX) bits are defined as follows:

Bits	Description
16	Action bit interpreted as follows: Bit Setting — Description 0 — recognize character 1 — inhibit character recognition
17-23	End of text character (ETX) to be recognized (7-bit ASCII character). The character must be different from STX, strip, ring, and terminate characters.

This character is recognized only during a receive blind order. Upon recognition, blind mode is set which blocks characters from being passed to memory. However, character recognition is still in effect for the STX and the ABC characters.

8. Termination CHAR A bits are defined as follows:

Bits	Description
24	zero
25-31	termination CHAR A (7-bit ASCII character) which must be different from STX, ETX, ring, and strip characters.

Recognition of CHAR A causes channel end if the character is recognized in the manner specified in bits 0-7.

9. Termination CHAR B bits are defined as follows:

Bits	Description
0	zero
1-7	termination CHAR B (7-bit ASCII character) which must be different from STX, ETX, ring, and strip characters.

Recognition of CHAR B causes channel end if the character is recognized in the manner specified in Word 3, bits 0-7.

10. Termination CHAR C bits are defined as follows:

Bits	Description
8	zero
9-15	termination CHAR C (7-bit ASCII character) which must be different from STX, ETX, ring, and strip characters.

Recognition of CHAR C causes channel end if the character is recognized in the manner specified in Word 3, bits 0-7.

11. Ring interrupt CHAR D bits are defined as follows:

Bits	Description
16	Action bit interpreted as follows: Bit Setting — Description 0 — recognize character 1 — inhibit character recognition
17-23	Ring interrupt CHAR D bits (7-bit ASCII character) which must be different from STX, ETX, strip, and terminate characters. When recognized, and no I/O is in progress, a ring interrupt is generated. Ring character reporting is enabled and disabled in the same manner as breaks.

12. Strip character (S1) bits are defined as follows:

Bits	Description
24	Action bit interpreted as follows: Bit Setting — Description 0 — recognize character 1 — inhibit character recognition
25-31	Strip character (S1) bits (7-bit ASCII character) which must be different from STX, ETX, ring, and terminate characters.

The S1 character is to be stripped from an incoming data stream. The character is unconditionally removed from the input buffer. The character cannot have been previously specified for a control function.

13. Strip character (S2) bits are defined as follows:

Bits	Description
0	Action bit interpreted as follows: Bit Setting — Description 0 — recognize character 1 — inhibit character recognition
1-7	Strip character (S2) bits (7-bit ASCII character) which must be different from STX, ETX, ring, and terminate characters.

The S2 character is to be stripped from an incoming data stream. The character is unconditionally removed from the input buffer. The character cannot have been previously specified for a control function.

Example

The following example shows a dial-in type 103 modem interface to an ALIM with one stop bit, even parity, parity enabled, 7-bit character length, baud rate factor of 16X, and data rate of 300 baud.

```
2003 MODEM REMOTE INIT 00001518 7A06CC00 0180800D 80800580 80000000
```

`00001518`

time-out of 90 seconds

`7A06CC00`

7	1 stop bit, even parity, parity enabled
A	7 bit character baud rate factor (16X)
06	300 baud rate
C	enable ring and break detection
C	switched line
00	must be zero

`0180800D`

01	terminate on CHAR A
80	STX characters – no action
80	ETX characters– no action
0D	CHAR A 0D=carriage return

`80800580`

80	CHAR B – no action
80	CHAR C – no action
05	ring-in character is <ctrl>E (X'05')
80	strip character 1 – no action

`80000000`

80	strip character 2 – no action
000000	must be zero

10.7.4 ACM/MFP Controller Record Syntax and Defaults

Syntax

ccaa [*baud*] [HALF|FULL] [ECHO|NOECHO] [*parity*] [*charsize*] [*stopbits*] [REMOTE] [NORTS] [MODEM] [GRAP] [NOTSM] [RXON] [WXON] [RHWF] [WHWF] [RDTR] [RRTS] [INIT *value*] [CXR=*n*] [LGC=*nnn*] [LGT=*sss*] [TMO=*mmm*]

ccaa is the channel and subaddress of the device.

[*baud*] is the baud rate: 19200, 9600, 7200, 4800, 3600, 2400, 2000, 1800, 1200, 600, 300, 150, 134, 110, 75, or 50. If not specified, the default is 9600 baud.

[HALF|FULL]
is the half or full duplex operation. If not specified, the default is HALF. Refer section 10.8.4.3, True Full-Duplex Operation for the ACM.

[ECHO|NOECHO]
specifies characters are or are not to be echoed by the computer as they are received. If not specified, the default is ECHO for FULL duplex or NOECHO for HALF duplex.

[*parity*] is ODD, EVEN, or NONE. If not specified, the default is EVEN. If NONE is specified, the serial character is smaller due to the absence of the parity bit.

[*charsize*] is the character size: 5, 6, 7, or 8. If not specified, the default is 7.

[*stopbits*] is the number of stop bits: S1, S1.5, or S2. If not specified, the default is S1.

[REMOTE] sets dial-up bit in UDT. If not specified, the default is not set.

[NORTS] resets the ready to send signal (RTS) after device initialization. This option allows software control of a peripheral switch controller in configurations with a controller present.

[MODEM] sets modem bit in UDT. If not specified, the default is not set. Also sets modem bit for ACE parameters. The default is no modem.

[GRAP] specifies graphic device. If not specified, the default is not graphic device.

[NOTSM] specifies a non-TSM device. If not specified, the default is TSM device.

[RXON] specifies software read flow control (XON/XOFF). If not specified, the default is no software read flow control. If this option is selected for a terminal, NOECHO should be specified so the terminal can be used in local echo mode. This avoids line contention caused by echoplex.

[WXON] specifies software write flow control (XON/XOFF). If not specified, the default is no software write flow control.

XON – no timeout value on writes
XOFF – 75 second timeout value on writes

[RHWF] specifies hardware read flow control (DTR). This parameter cannot be specified if either RXON or WXON has been specified. If not specified, the default is no hardware read flow control. If this option is selected for a terminal, NOECHO should be specified. This avoids line contention caused by echoplex.

[WHWF] specifies hardware write flow control (DTR). This parameter cannot be specified if either RXON or WXON has been specified. If not specified, the default is no hardware write flow control. If this option is selected for a terminal, the NOECHO parameter should be specified. This avoids line contention caused by Echoplex.

[RDTR] specifies the DTR line is used if RXON has been requested. This is the default, and it overrides RRTS if both are specified.

[RRTS] specifies the RTS line is used if RXON has been requested.

[CXR=*n*] *n* specifies the number of seconds to wait for carrier before J.TSM resets DTR, causing the modem to go on hook. Valid values for *n* are 1-255. The value specified by *n* must include delays introduced by the modem. The use of this option will also cause a 2 second delay at logon time before transmitting the logon banner.

[INIT *value*]

indicates the presence of hexadecimal initialization data. *value* is a 1-word hexadecimal value in the 8-line asynch initialization format. See ACM/MFP Initialization Format below.

[LGC=*nnn*]

nnn specifies the number of logon attempts, 0 to 255, allowed before the terminal is marked offline. If *nnn* is 0, there is no limit on the number of logon attempts. The default is 0.

[LGT=*sss*] *sss* specifies the number of seconds, 0 to 255, that a terminal waits at a logon prompt before being logged off. The default is 0.

[TMO=*mmm*]

mmm specifies the number of minutes, 0 to 999, that a terminal can remain I/O inactive.

10.7.4.1 ACM/MFP Initialization Format

	0	1	2	3	4	5	6 7	8	9	10	11	12 15	16 23	24 31
Word 1	Duplex	MR	FP	PS	PE	SB	CL	DC	CW	WD	DL	Baud rate	Wake-up character	Zero

Bit	Description
0	Full or half duplex bit is defined as follows: 0 full duplex 1 half duplex
1	Modem ring (MR) bit is defined as follows: 0 disable 1 enable
2	Forced parity (FP) bit is defined as follows: 0 normal parity as defined 1 force parity to one if odd; force parity to zero if even
3	Parity selection (PS) bit is defined as follows: 0 odd parity 1 even parity
4	Parity enabled (PE) bit is defined as follows: 0 disable parity 1 enable parity
5	Stop bit (SB) defined is as follows: 0 1 stop bit 1 2 stop bits or 1.5 stop bits for 5 character length
6-7	Character length (CL) is defined as follows:

Bit 6	Bit 7	CL
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

Bit	Description
8	ACM — Delta carrier (DC) bit is defined as follows: 0 disable attention on delta carrier 1 enable attention on delta carrier MFP — zero
9	ACM — Character write (CW) bit is defined as follows: 0 disable wait for the end of the last character write 1 enable wait for the end of the last character write MFP — zero

10 Wake-up detection (WD) bit is defined as follows:

- 0 allow wake-up character detection
- 1 inhibit wake-up character detection

11 Diagnostic loop (DL) bit defined as follows:

- 0 reset diagnostic loop
- 1 set diagnostic loop

12-15 Baud rate bits defined as follows:

Bit 12	Bit 13	Bit 14	Bit 15	ACM Baud rate	MFP Baud rate
0	0	0	0	50	75
0	0	0	1	75	110
0	0	1	0	110	134.5
0	0	1	1	134.5	150
0	1	0	0	150	300
0	1	0	1	300	600
0	1	1	0	600	1200
0	1	1	1	1200	2000
1	0	0	0	1800	2400
1	0	0	1	2000	4800
1	0	1	0	2400	1800
1	0	1	1	3600	9600
1	1	0	0	4800	19200
1	1	0	1	7200	38400
1	1	1	0	9600	External clock is 16X
1	1	1	1	19200	External clock is 1X

16-23 8-bit wake-up character

24-31 Must be zero

10.7.4.2 Example

The following example shows a dial-in modem interface to an ACM 8-line asynchronous with one stop bit, even parity, parity enabled, full duplex, 7-bit character length, and modem ring enabled.

```
7EC0 MODEM REMOTE INIT 5A050500
```

`5A050500`

5	full duplex, enable modem ring, normal parity as defined, even parity
A	parity enabled, one stop bit, 7-bit character
05	300 baud rate
05	ring-in character is <ctrl>E (X'05')
00	must be zero

10.7.4.3 True Full-Duplex Operation for the ACM

The ACM device can be used for true full-duplex operation. The device must have been SYSGENed using the device directive's FULL parameter. This creates two UDT's per full-duplex subchannel: one for the read sub-channel and one for the write subchannel. The write subchannel subaddress is the read subchannel address plus X'8'; for example, a subchannel address of 7E00 derives a write subchannel address of 7E08. The device must be initialized as NOECHO (local echoplex) to avoid I/O contention with the write subchannel. After the device is initialized, it is in full-duplex, single-channel mode; all read and write I/O is performed through the read subchannel.

To set the device to full-duplex, dual-channel mode, the TSM OPTION UNQUIET directive must be used. Dual-channel I/O reads are performed on the read subchannel; writes are performed on the write subchannel. Dual-channel mode is also entered by assigning the read and write subchannels separately, as if OPTION UNQUIET were in effect, and issuing an M.RELP (SVC 1,X'27') before any I/O is performed. I/O is then performed through the FCB associated with the appropriate subchannel. To reset to the single-channel mode, use the M.RESP (SVC 1,X'26') service.

10.7.5 Sample LOGONFLE

```
05                                            WAKEUP CHARACTER WRITE
*LGC=3                                        ! SYSTEM-WIDE MAX LOGON
                                                ATTEMPT OF 3
*LGT=60                                       ! SYSTEM-WIDE MAX LOGON TIMER
                                                OF 60 SECONDS
*TMO=60                                       ! SYSTEM-WIDE MAX TERMINAL
                                                TIMER OF 60 MINUTES
7EA0 19200 FULL                               ! DEFAULTS FOR OTHER VALUES
7EA1 9600 FULL NONE 8 S1 WXON REMO NOTSM      ! SAMPLE SERIAL PRINTER
7EA7 HALF MODEM REMOTE EVEN 300               ! DIAL-UP LINE
2000 9600 FULL EVEN 7                         ! ALIM
```

REMOTE	sets dial-up bit in UDT. The default is not set.
MODEM	sets modem bit in UDT. The default is not set. Also sets modem bit for ACE parameters. The default is no modem.
EVEN	even parity is used.
7	7 bit data is used.
FULL	full duplex operation.
NOTSM	specifies a non-TSM device. The default is TSM device.
HALF	half duplex operation.
8	8 bit data is used.
S1	specifies 1 stop bit is required.
WXON	device uses software write flow control (XON/XOFF).
LGC=3	specifies a system-wide default logon count of 3
LGT=60	specifies a system-wide default logon timer of 60 seconds
TMO=60	specifies a system-wide default terminal timer of 60 minutes

The following table shows the ACM configurations possible and their resulting operations.

SYSGEN Device Directive, FULL Option Specified	LOGONFLE FULL/HALF Option Specified	LOGONFLE ECHO/NOECHO Option Specified	Resulting Terminal Operation
yes	full	echo	half,echo
yes	full	noecho	full,noecho
yes	half	echo	half,echo
yes	half	noecho	half,echo
no	full	echo	half,echo
no	full	noecho	half,noecho
no	half	echo	half,echo
no	half	noecho	half,noecho

10.8 Using INIT

INIT is a TSM command. LOGONFLE is assigned for input by default.

Syntax

TSM>**$INIT** [*ccaa*]

[*ccaa*] specifies the appropriate channel and subaddress (hexadecimal) of a specific device to be reinitialized. The record from LOGONFLE that matches the channel and subaddress reinitializes the device.

If no device is specified, INIT uses the current version of LOGONFLE to reinitialize all devices that are currently free to allocate.

10.9 INIT Errors

INIT generates the following error messages:

`ATTEMPT TO MAKE TSM DEVICE A NON TSM DEVICE, ADDR=`*ccaa*

The NOTSM keyword is specified in LOGONFLE for a TSM device.

`DEVICE NOT PRESENT ADDR=`*ccaa*

The specified device is not plugged into the CPU.

`DEVICE NOT TERMINATED ADDR=`*ccaa*

The specified device is not plugged in on device end of line.

`INVALID LOGON COUNTER SUPPLIED.  DEFAULT IS 0.`

When setting up the defaults, an invalid character was entered for *nnn* in LGC=*nnn*. The default becomes 0.

`INVALID LOGON COUNTER SUPPLIED.  DEFAULT USED.`

When using a terminal-specific counter, an invalid character was entered for *nnn* in LGC=*nnn*. The system default is used.

`INVALID LOGON TIMEOUT VALUE SUPPLIED.  DEFAULT IS 0.`

When setting up the defaults, an invalid character was entered for *sss* in LGT=*sss*. The default becomes 0.

`INVALID LOGON TIMEOUT VALUE SUPPLIED.  DEFAULT USED.`

When setting up a terminal specific timeout value, an invalid character was entered for *sss* in LGT=*sss*. The default becomes 0.

`INVALID NON TSM DEVICE TYPE CODE, ADDR=`*ccaa*

Valid non-TSM device type codes are X'07' (CD) through X'0B' (PT) and X'0D' (CT) through X'1A' (U9).

`INVALID TERMINAL TIMEOUT VALUE SUPPLIED. DEFAULT IS 0.`

When setting up the defaults, an invalid character was entered for *mmm* in TMO=*mmm*. The default becomes 0.

`INVALID TERMINAL TIMEOUT VALUE SUPPLIED. DEFAULT USED.`

When setting up a terminal specific timeout value, an invalid character was entered for *mmm* in TMO=*mmm*. The default becomes 0.

`M.ASSN DENIAL,NO LOGON FILE DEFAULT USED`

A file named LOGONFLE is not on the system. Default parameters have been set. Non-TSM devices are not initialized.

`M.ASSN DENIAL,ADDR=`*ccaa*

The device at the specified channel and subaddress is in use and cannot be initialized.

`NO UDT ENTRY FOR ADDR=`*ccaa*

The specified device is not SYSGENed.

`NON TSM DEVICE WITHOUT NOTSM IN LOGONFLE, ADDR=`*ccaa*

A non-TSM device is specified in LOGONFLE without the NOTSM keyword.

`ON ADDRESS` *ccaa*`, FIELD UNIDENTIFIED:` *xxxxxxxx*

The string *xxxxxxxx* is not a valid keyword for a characteristic.

`TERMINAL SET-UP COMPLETE`

Initialization complete.

`WARNING, LOGON COUNTER VALUE TOO LARGE. CAPPED TO 255 ATTEMPTS`

The value *nnn* in LGC=*nnn* was larger than 255.

`WARNING, LOGON TIME VALUE TOO LARGE. CAPPED TO 255 SECONDS.`

The value *sss* in LGT=*sss* was larger than 255.

`WARNING, TERMINAL TIMEOUT VALUE TOO LARGE. CAPPED TO 999 MINUTES.`

The value *mmm* in TMO=*mmm* was larger than 999.

10.10 System Console Messages

Some system messages and operations pertain to the system console, and not to terminals running OPCOM, for example:

- mount and dismount messages for magnetic tape
- abort messages and codes for tasks running in the real-time environment
- I/O error conditions that can either be corrected offline with I/O resumed or aborted

Displaying these messages on the console assumes a central location of configured system hardware, including the system console, the CPU, printers, and tape units. User tasks can also send messages to the system console.

10.10.1 Information Messages

Commands or responses to prompts at the system console always take precedence over messages sent to the console. The maximum output string that cannot be interrupted for a message is 72 characters.

10.10.2 Action Messages

If a system message requires operator intervention and reply, the condition noted must be addressed. The operator enters a reply and terminates it with a carriage return. For example,

```
*CR7800 INOP : R,A? R
```

A card reader is not operating. The operator fixes it and responds `R` and a carriage return to resume. Input continues.

10.10.3 Terminal Messages

The terminal user receives only the OPCOM messages that pertain to directives issued from that terminal.

10.11 Floppy Disk Media Initialization (J.FORMF)

All IOP floppy disks must be properly formatted before they can be used on the operating system.

To format a floppy disk, type J.FORMF at the TSM prompt. The following message is displayed:

```
J.FORMF-SPECIFY FORMAT:MODE0/MODE1/QUIT (REPLY 0, 1, OR Q):
```

If `0` (MODE0) is entered, the floppy disk is formatted as double density, 256 bytes per sector. The first physical sector of all tracks starts at hexadecimal 0 and ends at hexadecimal 19.

If `1` (MODE1) is entered, the floppy disk is formatted as double density, 256 bytes per sector. The first physical sector of all tracks starts at hexadecimal 1 and ends at hexadecimal 1A. MODE1 formatting allows data files to be transported to other computer systems that expect the physical sectors to begin at hexadecimal 1 and end at hexadecimal 1A. MODE1 is only supported on systems containing a Model 8031 Line Printer/Floppy Disk Controller.

If `Q` (QUIT) is entered, control returns to TSM and the floppy disk is not formatted.

The default mode selection should be MODE0 and must be MODE0 if data files are to be transported to another Encore computer containing either a Model 8030 or 8031 Line Printer/Floppy Disk Controller.

The default device for J.FORMF is device address FL7EF0. To override the default, specify the following assign directive at the TSM prompt before J.FORMF is activated:

```
$ASSIGN FL TO DEV=FLxxxx BLO=N
```

xxxx is the drive device address to be used by J.FORMF.

10.12 M.MOUNT File

The M.MOUNT file automatically mounts volumes at IPL. It can be used to mount both public and nonpublic volumes. (A nonpublic volume requires a logical mount by subsequent users to access the resources on that volume). Use of the nonpublic mount should be determined by the specific needs of the system applications.

The M.MOUNT file directive syntax is the same as the OPCOM MOUNT directive. Refer to the OPCOM section in the MPX-32 Reference Manual, Volume II. The file must be stored unnumbered on the SYSTEM volume.

Examples

```
EDT>COL
1.  MOUNT ATLAS04 ON DM0804 OPTION=PUBL
2.  MOUNT TB02 ON DM0802 OPTION=PUBL
     .
     .
     .
5.  <ret>
EDT>STO M.MOUNT SYS UNN
```

If desired, an asterisk (*) can be used in the volume name field. This causes the volume on the specified drive to be mounted, regardless of its name.

At system initialization, a mount message is displayed for each volume in the file. If an ABORT (A) is entered to any message, an attempt is made to MOUNT any remaining volumes.

10.13 Operator Intervention Inhibit

When operator intervention is inhibited, all prompts normally displayed on the system console are suppressed and the following defaults are in effect:

Prompt	Default
system date/time	more recent date/time recorded by J.DTSAVE or last system volume mount
swap volume channel and subaddress	system volume
mount messages – formatted volumes	ready
dual-port volume cleanup confirmation*	no
wait I/O request retry	abort (DP0)
invalid port ID specification*	port zero
dual-port volume mount request verification for an allocated port*	continue

The prompts marked by an asterisk (*) apply only when a dual-processor shared volume is mounted. If the default responses to these prompts are not the desired responses, unreliable file system management can result.

After system initialization, wait I/O request retry prompts remain inhibited. When a wait I/O retry is requested, a message is displayed on the system console specifying the inoperable device and the request is aborted.

Mount messages are not inhibited after system initialization. The SYSGEN or OPCOM MODE directive inhibits mount messages after system initialization. If mount messages are inhibited and a device is not ready when J.MOUNT issues a mount request, the request is repeated approximately 60 seconds later. If the device is still not ready after the second request, the mount request is aborted.

When operator intervention is inhibited and an attempt is made to boot a master system image, operator intervention inhibit is overridden and an operator must respond to the prompts.

10.14 System Date/Time Backup Program (J.DTSAVE)

The System Date/Time Backup Program (J.DTSAVE) is a privileged, nonresident system program. J.DTSAVE records the current date and time to the system volume at regular intervals. The intervals are established by the SYSGEN directive DTSAVE. If operator intervention is inhibited, the system date and time are initialized to either the last date and time recorded by J.DTSAVE or the date and time of the last system volume mount, whichever is more recent. Entering a chronologically accurate date increases the reliability of file system management. J.DTSAVE is activated by the SYSGEN and OPCOM ACTIVATE directives.

10.15 System Shutdown

MPX-32 offers a system shutdown capability to increase reboot performance. The system administrator can request the following types of shutdown:

Shutdown type	Description
immediate forced	System shuts down immediately, without waiting for user and task activity to end.
timed forced	System waits a specified time for user and task activity to end, then shuts down when this activity ends or time expires.
indefinite	System waits indefinitely for all user and task activity to end before it shuts down.

Shutdown ends with a system halt or, if requested, with an online restart. This restart can be to the default system image or to a specified image.

The system administrator requests a shutdown by using either the J.SHUTD task or a shutdown macro. These differ as follows:

- J.SHUTD task performs a shutdown without notifying users that a shutdown is in progress. This task deallocates system resources, requests volume dismounts, and halts or restarts the system.
- A shutdown macro provides an interface to users and application tasks during shutdown. It informs users when a shutdown is in progress, ends application tasks specified in the macro, and calls J.SHUTD to perform the shutdown. A sample macro, SHUTDOWN, is included on the Master SDT and can be customized for any system.

Note: To shutdown a system with active users, we recommend using a shutdown macro. Use the J.SHUTD task only when the system has no active users.

10.15.1 Using J.SHUTD

For a timed or indefinite shutdown, J.SHUTD requests the dismount of all volumes and sends a message to tasks holding resources on the system volume. When all system volume resources are deallocated, J.SHUTD marks the volume as quiescent and physically dismounts it if the current image is the default. Other volumes are physically dismounted as their resources are deallocated. When the dismounts complete and the system volume is quiescent, or when time expires, J.SHUTD halts the system. If a restart is requested, J.SHUTD restarts to the default or specified image.

For an immediate forced shutdown, J.SHUTD halts the system instantly, without waiting for volume dismounts or for the system volume to be quiescent. If requested, J.SHUTD restarts the system.

Syntax

J.SHUTD [[**TIME=**]*ss*] [**RESTART** [*image*]]

[**TIME=**]*ss* specifes the number of seconds for J.SHUTD to wait for system activity to end before forcing a shutdown. Specify 0 to shutdown immediately. Omit the option to initiate an indefinite shutdown.

RESTART requests an online restart following shutdown. If this option is omitted, shutdown ends in a system halt.

image is the file name of the system image to restart. When an image is specified, MPX-32 cleans up the system volume. If omitted, the default image is used.

After J.SHUTD is called, system shutdown cannot be canceled.

10.15.2 Shutdown and Volume Cleanup

MPX-32 cleans up the system volume after shutdown in the following cases:

- shutdown is forced before the system volume dismounts
- the image requested for restart is not the current image
- the current image before shutdown is not the default image
- a different image is made the default image without restarting

10.15.3 SHUTDOWN Macro

The shutdown macro, SHUTDOWN, notifies users of a pending shutdown and initiates this shutdown by calling J.SHUTD. This macro can be modified to set how often it sends a message to users, which tasks it deallocates or removes before shutdown, and what value it passes to the J.SHUTD TIME option. The macro can pass up to four parameters to J.SHUTD. These parameters must correspond to the J.SHUTD syntax or an error results.

Shutdown can be canceled while the macro is running by using a $REMOVE command to remove the job.

10.15.4 Using SHUTDOWN

The SHUTDOWN macro can be run either in batch or interactive mode. To avoid tying up a terminal during shutdown, we recommend you batch the macro or set the terminal page size to zero ($PAGESIZE 0) before running the macro interactively.

Syntax

BATCH SHUTDOWN [[TIME=] *ss*] [RESTART[=*image*]]

[**TIME=**]*ss* specifies the number of seconds before forcing a shutdown. Specify 0 for an immediate forced shutdown. Omit a value to initiate an indefinite shutdown. The macro uses ss in calculating when to call J.SHUTD.

RESTART requests a restart following shutdown. The macro passes this parameter to J.SHUTD.

image specifies the filename of the system image to restart. The macro passes this parameter to J.SHUTD.

To run the macro interactively, omit the BATCH keyword from the syntax above.

The macro can also be batched from a program through the TSM procedure call service (M.TSMPC/M_TSMPC). Refer to the MPX-32 Reference Manual, Volume I, Chapter 6 or 7 for information on the M.TSMPC or M_TSMPC service.

10.15.5 Modifying the SHUTDOWN Macro

To control the time between signals to users, the macro uses the PAUSE task. Each call to the task has the following syntax:

PAUSE *n*

n is the number of seconds the task pauses until the next signal to users.

Each call to PAUSE specifies a value (*n*) for the number of seconds to pause. The macro uses smaller *n* values as the time remaining to shutdown decreases. These *n* values can be modified throughout the macro to increase or decrease the pause between signals.

To determine when to call J.SHUTD to perform the shutdown, the macro uses the ACTIME constant. This constant is the number of seconds (less than 60) that the macro passes to the TIME option of J.SHUTD. The macro runs for *ss* minus ACTIME seconds before calling J.SHUTD. When the macro is batched with *ss* less than or equal to 60, it resets ACTIME to *ss*. This ACTIME constant can be modified to increase or decrease the value passed to the J.SHUTD TIME option.

The macro signals users based on ACTIME and the macro's TIME=*ss* option as follows:

- If *ss* is omitted, the macro informs users that an indefinite shutdown is in progress. It calls J.SHUTD to handle the shutdown and the macro ends.
- If *ss* is 0, the macro informs users that the system is shutting down immediately. It calls J.SHUTD to halt the system and the macro ends.
- If *ss* is 60 seconds or less, the macro informs users that the system is shutting down in less than 1 minute. It calls J.SHUTD and passes *ss* to the J.SHUTD TIME option.

- If *ss* is less than 1 hour and greater than 1 minute, the following table applies:

Minutes until shutdown	Macro signals every
1-3	1 minute
4-5	3 minutes
6-20	5 minutes
21-60	15 minutes

  When 1 minute remains, the macro sends a final message and pauses for 60 seconds minus ACTIME, then calls J.SHUTD.

- If *ss* is greater than 1 hour and less than 2 hours, the macro informs users that the system is shutting down in the given time and it pauses for *ss* minus 1 hour. It then sends another message informing users of the time remaining to shutdown. When 1 hour remains, the table above applies.
- If *ss* is greater than or equal to 2 hours, the macro informs users that the system is shutting down in the given time and it pauses for 1 hour plus the number of minutes and seconds in the given time. For example, if TIME=7300 seconds (2 hours, 1 minute, 40 seconds), the macro pauses for 1 hour, 1 minute, 40 seconds before sending another message. The macro then sends a message every hour until 1 hour remains and the table above applies.

10.15.6 Error Messages

The following messages are written to LFC ERR:

```
J.SHUTD: SYSTEM ADMINISTRATOR ATTRIBUTE IS REQUIRED
J.SHUTD: IMAGE NAME MUST BE 0 TO 8 CHARACTERS
J.SHUTD: TOO MANY ARGUMENTS OR INVALID ARGUMENT SUPPLIED
```

The following messages are written to the console only:

`J.SHUTD: DISMOUNT REQUEST TO VOLUME` *name* `RETURNED WITH RM`*nn*

name is the name of the volume that J.SHUTD requested to dismount

nn is a REMM abort code. Refer to Appendix C for a list of these abort codes or use the TSM $ERR directive.

Note: J.SHUTD returns an RM14 error indicating that dismount is pending if the volume still has resources allocated.

`J.SHUTD: RUN REQUEST TO RESTART RETURNED FROM M.SRUNR WITH STATUS VALUE` *mm*

mm is a status value returned from the call to M.SRUNR. Refer to the M.SRUNR system service in Chapter 6 of MPX-32 Reference Volume I for a list of these status values and their meaning.

If a restart or request for restart fails, halt and reboot the system.

The SHUTDOWN macro sends the following messages to owner SYSTEM:

```
SHUTDOWN: INVALID ARGUMENT SPECIFIED. JOB TERMINATED.
SHUTDOWN: TIME ARGUMENT IS INVALID. JOB TERMINATED.
SHUTDOWN: IMAGE NAME GIVEN IS INVALID. JOB TERMINATED.
```

Note: If the SHUTDOWN macro parameters are entered incorrectly, TSM issues an INVALID PARAMETER TYPE message.

10.16 Swap Scheduler Control Options

The swap scheduler processes entries in the memory request queue (MRQ). It provides memory allocation and swap scheduling for individual memory requests. So that the swap scheduler can function according to the requirements of a system, the following swap scheduler options can be set by the system administrator:

1. Swapper algorithms
2. Wait state ordering
3. Wait state swap-on priority only
4. Call back swap-on priority only
5. User set swap-on priority only flag
6. User set swap inhibit flag
7. Swap thrash control
8. Task group outswap limits

To set or reset an option, edit the Swap Option File (SJ.SWAPR2). SJ.SWAPR2 contains instructions on how to edit the file. After the file is edited, it must be resaved as SJ.SWAPR2. Then, the JCL file, JJ.A.SWP, must be run to catalog the swapper load module (J.SWAPR). After J.SWAPR is cataloged, the system administrator must restart the system to incorporate the modified swap scheduler into the operating system.

If SJ.SWAPR2 is not modified, the swap scheduler options are defaulted so that J.SWAPR performs similar to MPX-32 Revision 3.2C. To revert to swapping that is similar to MPX-32 Revision 3.2C after the swap option file has been edited, re-edit the SJ.SWAPR2 file and set all options to their default settings.

Swap-on priority only (SOPO) means that a task will be outswapped only when memory is required for an equal or higher priority task, regardless of the task's state. If swap-on priority only is not in effect, a higher priority task in any wait state can be outswapped for a lower priority task that is ready to run. If the outswapped higher priority task changes to a ready-to-run state too soon, the task is inswapped again causing delays and reducing system performance.

10.16.1 Swapper Algorithms

There are two swap scheduler algorithms:

Algorithm 1 — determines if sufficient swappable memory is available for the memory requestor before selecting an outswap candidate. If Algorithm 1 is set and the requested memory is not available, the swap scheduler suspends until the next memory scheduler event.

Algorithm 2 — outswaps a task without determining if sufficient swappable memory will be available for the memory requestor. If no memory is outswapped for one second, the outswapped tasks are eligible to be inswapped. When there has not been an inswap for one second, the cycle can repeat. This algorithm assumes that the highest priority task on the memory request queue will receive its required memory.

To set this option, edit the Method section of SJ.SWAPR2.

10.16.2 Wait State Ordering

The Wait State Ordering option determines the order that the swap scheduler searches the queues for swap candidates. The Swap Option File (SJ.SWAPR2) contains a list of queues that can be arranged in any order. If this list is not modified, the swap scheduler searches the list as follows:

Queue	Order
HOLD	first
SUSP	
RUNW	.
SWDV	
SWDC	.
SWSR	
SWSM	.
SWLO	
SWFI	.
MRQ	
ANYW	.
SWGQ	
SWTI	.
SWIO	
SWMP	last

To change the list, edit the Wait State Search section of SJ.SWAPR2.

Caution: All 15 wait states must appear in the list.

10.16.3 Wait State Swap-on Priority Only (SOPO)

The Wait State Swap-on Priority Only (SOPO) option allows any of the 15 wait states to be Swap-on Priority Only. This option allows all tasks in the designated wait state to be SOPO. If this option is not set, tasks are not designated SOPO because of their wait state.

To set this option, edit the Wait State Search section of SJ.SWAPR2.

10.16.4 Call Back Swap-on Priority Only (CB.SOPO)

The Call Back Swap-on Priority Only (CB.SOPO) option determines whether a task is SOPO when callback is pending for a no-wait message request or a no-wait run request. If this option is not set, the task is not SOPO.

To set this option, edit the CB.SOPO section of SJ.SWAPR2.

10.16.5 User Set Swap-on Priority Only Flag (US.SOPO)

The User Set Swap-on Priority Only Flag (US.SOPO) option allows a privileged task to set a flag bit in the DQE (DQE.USPO). This allows the task to be SOPO until the bit is reset. If this option is not set, DQE.USPO is ignored.

To set this option, edit the US.SOPO section of SJ.SWAPR2.

10.16.6 User Set Swap Inhibit Flag (US.SWIF)

The User Set Swap Inhibit Flag (US.SWIF) option allows a privileged task to set a flag bit in the DQE (DQE.USWI). This specifies the task as ineligible for outswapping until the bit is reset. If this option is not set, the swap inhibit bit (DQE.USWI) is ignored.

To set this option, edit the US.SWIF section of SJ.SWAPR2.

10.16.7 Swap Thrash Control

Swap thrashing may occur when tasks that are frequently changing between the ready-to-run and wait states are competing for memory. Thrashing causes these tasks to be swapped between memory and the swap file on the disk as their state changes. To reduce swap thrashing and improve system performance, the user can set SWAPDUTY or SWAPPERI.

SWAPDUTY is a value between 0 and 100 that indicates the percentage of time that the swap scheduler can actively swap tasks. If the swap scheduler activity is greater than SWAPDUTY, J.SWAPR processes only swap-on priority only tasks. If the SWAPDUTY is 100, there is no thrash control and all tasks are swapped according to wait state. If the SWAPDUTY is 0, only SOPO tasks are swapped. If most of the tasks are SOPO and the swap scheduler is very active, thrashing may occur.

SWAPPERI is a value between 2 and 64 seconds. If set to a value less than 2, SWAPPERI defaults to 2 seconds. If set to a value greater than 64, SWAPPERI defaults to 64 seconds.

The percentage of the swap activity is constantly being monitored by the system. If the swap activity is greater than the SWAPDUTY, only SOPO tasks are swapped. If the percentage of swap activity is less than the SWAPDUTY, all tasks are swapped on a wait state basis.

If this option is not set, the system default is no thrash control and SWAPDUTY is measured every 2 seconds per megabyte of memory.

To set this option, edit the Thrash Control section of SJ.SWAPR2.

10.16.8 Task Group Outswap Limits

The Task Group Outswap Limits option provides performance predictability for task swapping by controlling memory availability on a task group basis. This is accomplished by assigning tasks to task groups and specifying the outswap limit (the amount of memory eligible for outswap) for each task group.

To improve system throughput and reduce the chance of memory deadlock, the Task Group Outswap Limits option can override the outswap limit. This is done only when the memory requestor and outswap candidate are in the same task group. In this situation, the Task Group Outswap Limits option reallocates memory from the outswap candidate to a different task in the task group. It then outswaps the outswap candidate and inswaps the memory requestor.

To specify the Task Group Outswap Limits option, complete the following:

- Configure the system for type S memory with the SYSGEN SIZE directive. For more information, refer to Chapter 7, the System Generation (SYSGEN) chapter, in this volume.
- Set the Swapper Algorithms option for Swap Algorithm 2.
- Set the task group outswap limit in the outswap limit table by editing SJ.SWAPR2.
- Assign a task group identification number to the task with the Cataloger CATALOG directive. For example, the following CATALOG directive statement assigns TASKA to task group five:

  ```
  CATALOG TASKA TGID=5
  ```

  Cataloger then stores the task group identification number, in this case five, in the load module preamble (byte 0 of word 45) for the task.

Notes:

A task can be assigned to only one task group.

If the TGID parameter is not specified in the CATALOG directive, the task group identification number defaults to 0.

Up to 16 task groups can be assigned.

System deadlock can occur if memory requirements cannot be met because the outswap limits force a task to become unswappable.

10.16.8.1 Errors

If there is not enough memory for an inswap of the highest priority task on the memory request queue (MRQ), an error occurs. The following message displays on the terminal:

```
**** WARNING: SWAPPER ANOMALY HAS OCCURRED.
NOT ENOUGH MEMORY FOUND FOR THE HIGHEST PRIORITY TASK IN THE MRQ.
```

This message displays on the terminal at timed intervals that can be set with the MSGTIME parameter in SW.SWAPR2.

10.17 Multiprocessor Recovery Task (J.UNLOCK)

J.UNLOCK is the multiprocessor recovery task. In a multiprocessor system, J.UNLOCK allows a processor to recover and continue processing when another processor goes offline. Resource locks, assign counts, and user counts owned by the offline processor are removed from the shared volumes by the online processor.

J.UNLOCK is a resident, privileged task. It is activated by an OPCOM UNLOCK directive or by J.MOUNT when a multiprocessor shared volume is mounted. When the last multiprocessor shared volume is dismounted from the system, J.UNLOCK deactivates.

10.18 Shadow Memory

After the shadow memory board is installed and shadow memory is configured, the run-time performance improves for a task that uses shadow memory.

MPX-32 supports three shadow memory boards which can be configured from address 0 to the end of memory. The 128 KB board must be configured at multiples of 16 map blocks, the 512KB board at multiples of 64 map blocks, and the 4MB board at multiples of 512 map blocks. The following shadow memory configurations are supported.

- a single processor (IPU or CPU) with a single region of shadow memory. See Figure 10-1.
- two processors (IPU and CPU) with a single region of shadow memory. The regions must have the same starting and ending addresses and be shadowed by an equal number of shadow memory boards on both processors. See Figure 10-2.
- two processors (IPU and CPU) with two nonoverlapping shadow memory regions. The starting addresses of both regions are different and the number of shadow memory boards for each processor can be unequal. See Figure 10-3.

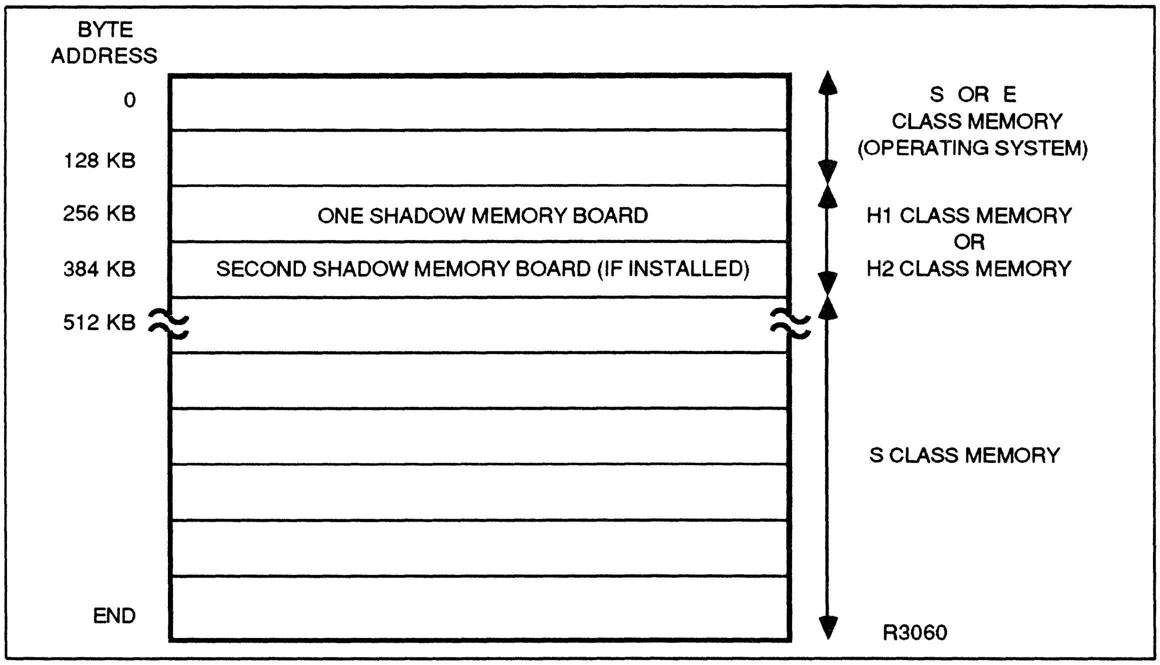

Figure 10-1
Shadow Memory Configuration with a Single Processor and a Single Region of Shadow Memory Using 128 KB Shadow Board

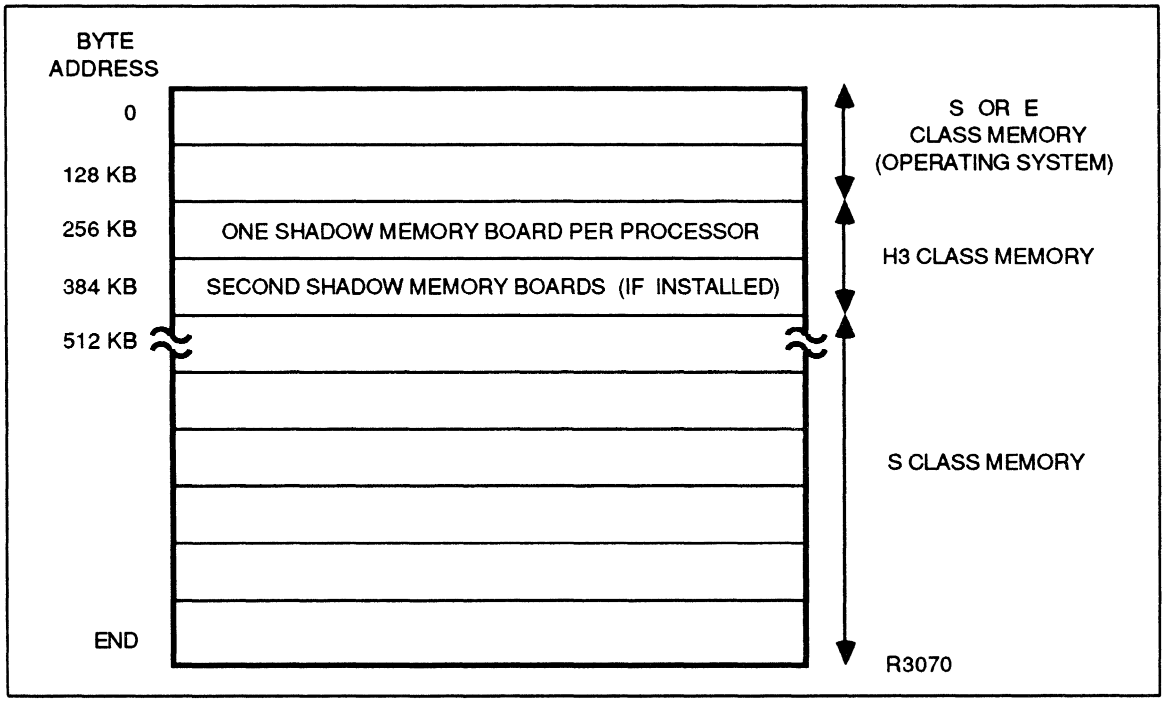

Figure 10-2
Shadow Memory Configuration with Two Processors and a Single Region of Shadow Memory Using 128 KB Shadow Board

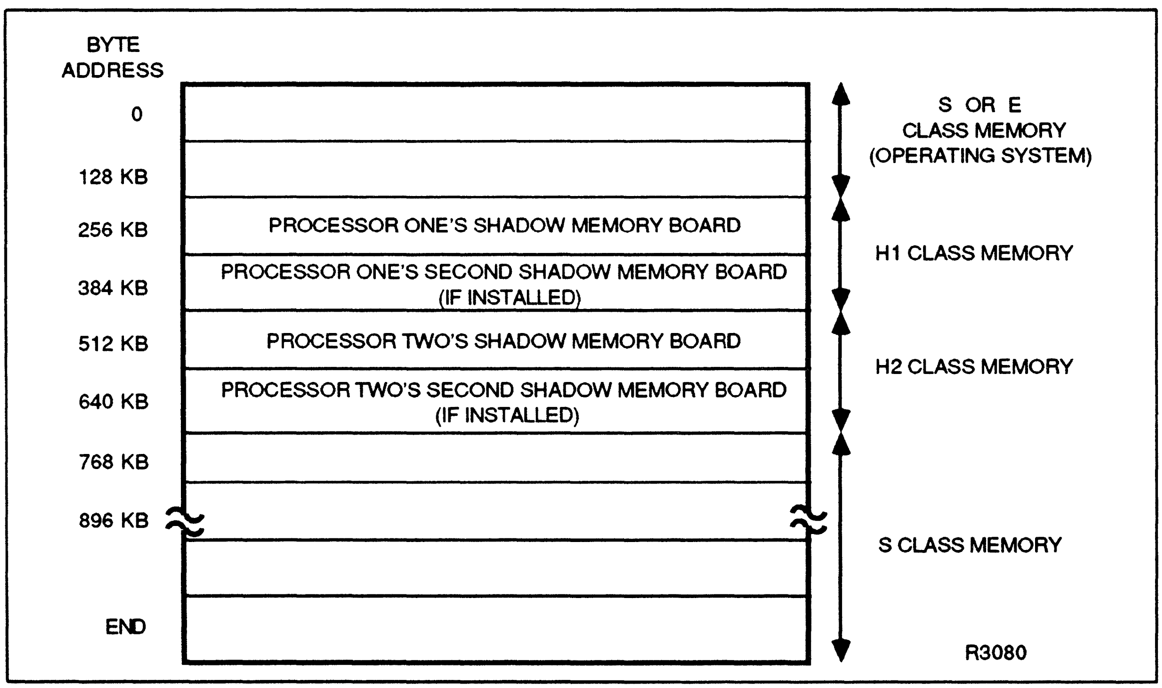

Figure 10-3
Shadow Memory Configuration with Two Processors and Two Regions of Shadow Memory using 128 KB Shadow Board

Configurations not supported are:

- processors with multiple shadow boards that are not jumpered over a contiguous region of physical memory

A shadow memory board can be jumpered to physical address zero which causes the resident portion of MPX-32 to be shadowed.

A processor's shadow memory is assigned to a contiguous region in physical memory; shadow memory is not supported for noncontiguous regions of physical memory.

There are three shadow memory classes:

H1	Shadow memory installed for processor 1.
H2	Shadow memory installed for processor 2.
H3	Processor 1 and 2 have equal amounts of shadow memory, and are both jumpered to the same addresses.

These classes specify the memory types used in the SYSGEN TYPE directive. The TYPE directive can reserve shadow memory for tasks that request shadow memory.

Notes:

The Processor Select Switch determines which processor is the CPU and which is the IPU. MPX-32 matches the processors to the CPU or IPU and determines if shadow memory is present.

If shadow memory is not reserved, tasks that require S class memory can allocate shadow memory. This can cause tasks that require the shadow memory to wait for shadow memory. If a task requests shadow memory and shadow memory has been allocated, J.SWAPR attempts to allocate shadow memory according to the task's state and priority.

The difference between H class memory and the shadow memory classes is that normally, if H class memory is not available, E class can be allocated. With the shadow memory classes, E class memory is never requested for the shadow memory area. H class memory cannot be used with H1, H2, or H3 class memory.

After a task has been configured, there are three ways that a task can access shadow memory:

J.SHAD — This utility permanently specifies the portions of a task's logical address to be located in shadow memory. For more information on J.SHAD, refer to MPX-32 Reference Volume II.

$SHADOW — This directive specifies the portions of a task's logical address to be located in shadow memory. For more information on $SHADOW, refer to MPX-32 Reference Volume II.

M.PTSK — This service activates a task. When used with a type 11 RRS (assign to shadow memory), it specifies the portions of a task's logical address to be located in shadow memory. Nonshadow memory is specified as E or S class in word 0, byte 3 of the parameter block. This service is only for privileged users. For more information on M.PTSK, refer to MPX-32 Reference Manual Volume I.

10.19 Error Messages

If shadow memory is not software configured and physical shadow memory is present and noncontiguous, the following message appears on the console:

```
NOTE: NO SHADOW MEMORY SYSGENED AND NONCONTIGUOUS {CPU|IPU} SHADOW
MEMORY DETECTED
```

If shadow memory is software configured and physical shadow memory is present and noncontiguous, the following message appears:

```
NOTE: NONCONTIGUOUS {CPU|IPU} SHADOW MEMORY DETECTED - RUNNING IN
DEGRADED MODE
```

If shadow memory is not software configured, the image is not a master image, and physical shadow memory is present but does not cover all the memory that is software configured, the following message appears:

```
NOTE: NO SHADOW MEMORY SYSGENED BUT {CPU|IPU} SHADOW MEMORY DETECTED
```

When shadow memory is software configured and configuration errors are detected because physical shadow does not match that configured, the following message appears:

```
WARNING: {CPU|IPU} SHADOW MEMORY ERRORS - MAY BE RUNNING IN DEGRADED
MODE
```

The following display occurs after any of the above error messages:

```
{CPU|IPU} SHADOW MEMORY size BOARD
UNITx AT {nnnnnnnn | MALFUNCTION OR NONPRESENT}
```

size	indicates the size of the shadow memory board: 128KW, 512KW or 4MB
x	specifies the unit number
nnnnnnnn	specifies the address of the shadow memory board

10.20 Memory Disk

The MPX-32 memory disk is a partition of main memory. The memory disk simulates a moving head disk, except the memory disk:

- seeks and accesses data at a faster rate
- has a lower capacity
- is not bootable
- cannot be used as a physical cache disk
- can only be single or dual-ported

If configured, the memory disk partition is allocated during system initialization, unless the system administrator requests otherwise at system generation.

There are two system tasks for memory disk usage — J.MDSAVE and J.MDREST. J.MDSAVE dumps the contents of memory disk to a system file. J.MDREST restores any unmounted memory disks saved by J.MDSAVE. See the J.MDSAVE and J.MDREST sections in this chapter for more information.

Memory disk can be marked online and offline by the OPCOM ONLINE and OFFLINE directives. These directives also allow the allocation and deallocation of the memory disk's memory. See OPCOM ONLINE and OFFLINE in MPX-32 Reference Manual Volume II for more information.

The memory disk handler is named H.MDXIO. Refer to the MPX-32 Technical Manual Volume II for more information.

Notes:

Dual-ported memory disks can be located only in a multiprocessor shared memory area. Single-ported memory disks can be located in any memory area except the multiprocessor shared memory area.

Up to 8 memory disks can be configured as long as the subaddress is not the same as the null device. Memory disks can be configured on channel 00, subaddress 0 and any following even subaddress. Memory allocated to a memory disk must be contiguous.

If the reloading of all memory disks is required at restart or IPL, use the SYSGEN SEQUENCE directive to activate J.MDREST. The M.MOUNT directive cannot reload a memory disk and should not be used to mount any memory disks that will be reloaded.

High priority real-time tasks that relinquish control only while I/O is in progress take an unfair amount of CPU time unless their I/O priorities are modified.

10.20.1 Memory Disk Configuration

The configuration size of a memory disk must be greater than 12KB. The memory disk size is specified at SYSGEN. To configure a memory disk, the following steps are necessary:

1. Add a SYSGEN CONTROLLER directive for each memory disk (channel 00).
2. Add a SYSGEN DEVICE directive for each memory disk.
3. Add a SYSGEN SEQUENCE directive for J.MDREST, if required. For more information on this option, refer to the J.MDREST system task in this chapter.
4. Perform a SYSGEN.
5. Restart the system.

Refer to the appropriate directive for specific memory disk information.

10.20.2 Memory Disk Usage

10.20.2.1 Formatting and Mounting a Memory Disk

After a system restart, a memory disk is formatted using the Volume Formatter (J.VFMT). A memory disk can then be mounted using the OPCOM MOUNT or TSM $MOUNT directive.

10.20.2.2 Accessing a Memory Disk

After a memory disk has been formatted and mounted, it can be accessed like a hard disk; for example, through Volume Manager and the Text Editor.

10.20.2.3 Dismounting a Memory Disk

If a memory disk is mounted with the OPCOM MOUNT directive, it must be dismounted with the OPCOM DISMOUNT directive. The same is true for the TSM MOUNT and DISMOUNT directives. Failure to specify the corresponding directive results in an error message.

10.20.3 Memory Disk Aborts and Errors

10.20.3.1 Abort Cases

The following is the memory disk abort code and its description.

`MM01`	Request for memory disk I/O to a location outside the memory disk boundaries

Aborts during memory disk installation occur:

- if DM00 is not specified in the CONTROLLER directive for a memory disk — SYSGEN aborts, and an error message is generated in the SYSGEN listed output file
- when reloading a memory disk that has been publicly mounted with M.MOUNT
- during an attempt to mount or format a memory disk for which no partition is present

Other possible aborts are the same as for other types of disk drives.

10.20.3.2 Errors

Errors that result from improper configuration cause a trap at assign time. The following memory disk errors result in messages displayed on the operator's console:

- insufficient contiguous memory of the correct type
- requests for OPCOM to mark online a deallocated memory disk that cannot be fulfilled because there is no available contiguous memory
- attempts to deallocate a dual-ported memory disk's memory

All other errors, such as end-of-medium and full disk, are a result of improper disk usage, and display the same error messages as other types of disk device errors.

10.20.4 Memory Disk Save Task (J.MDSAVE)

J.MDSAVE saves the contents of a memory disk to a system file. To activate J.MDSAVE, logon as the system administrator, then type:

`J.MDSAVE DM00`*`nn`*

nn is the subchannel of the memory disk

The contents of the specified memory disk are saved to a system file named @SYSTEM(SYSTEM)DM00*nn*. See Figure 10-4.

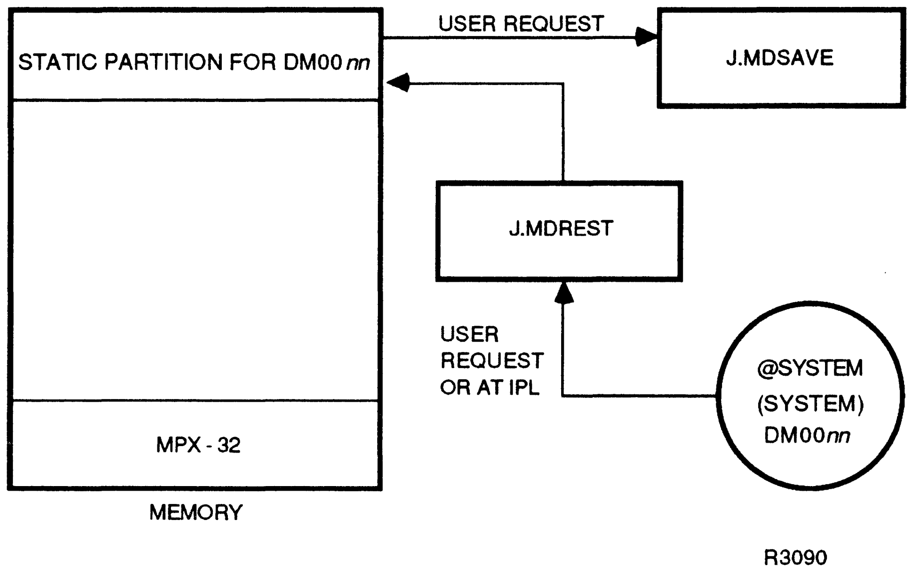

Figure 10-4
Memory Disk

10.20.5 Memory Disk Restore Task (J.MDREST)

J.MDREST restores all unmounted memory disks that have been saved by the J.MDSAVE system task. Before using J.MDREST, the J.MDSAVE system task must have been used. See Figure 10-4. To activate J.MDREST, logon as the system administrator, then type:

```
J.MDREST
```

When the SYSGEN SEQUENCE directive is entered as follows, J.MDREST is activated at system initialization:

```
SEQUENCE=J.MDREST
```

This reloads any unmounted memory disks from the system disk files saved by J.MDSAVE (@SYSTEM(SYSTEM)DM00*nn*). It is important to activate J.MDREST with the SYSGEN SEQUENCE directive and not the SYSGEN ACTIVATE directive. If ACTIVATE is used, an error message is displayed.

The M.MOUNT file should not contain entries for the memory disks to be reloaded. If the M.MOUNT file contains entries for memory disks to be reloaded, the mount messages for these entries should be aborted. If MODE=SNOP is specified in the SYSGEN MODE directive, no operator messages are displayed and the disks are mounted.

10.21 Label ANSI Tape Utility (J.LABEL)

The Label ANSI Tape Utility (J.LABEL) writes the initial header labels to new ANSI tapes. All ANSI tapes must be labeled by the J.LABEL utility to be recognized by the system. This utility can be used only by the system administrator. For J.LABEL syntax and usage information, refer to the ANSI Labeled Tape Utilities in MPX-32 Reference Manual Volume II.

11 Device Initializer/Loader (DEVINITL)

11.1 Introduction

The Device Initializer/Loader (DEVINITL) initializes devices by using an initialization directive file. DEVINITL also loads microcode onto devices having Writable Control Storage (WCS) capabilities by using a firmware file in conjunction with the initialization directive file.

11.2 Initialization Directive File

DEVINITL directives are in a file named DEV_INIT located on the system volume and system directory. This file contains default directives used when DEVINITL is activated by TSM. The DEV_INIT file contains the following three types of directives:

- Option Specification (optional) — overrides default parameters for file matching and message displays.
- Device Load (required) — specifies devices to be initialized and optionally specifies files that contain microcode to be loaded.
- Device Initialization (optional) — specifies a directive byte and/or data required to initialize a device.

Based on these directives and their defaults, if any, DEVINITL initializes devices and/or loads microcode, checking only record type, length, and checksum. Messages are displayed indicating successful device initialization and/or loading or the errors which occurred preventing successful initialization and/or loading.

The format of the initialization directive file is shown in Figure 11-1.

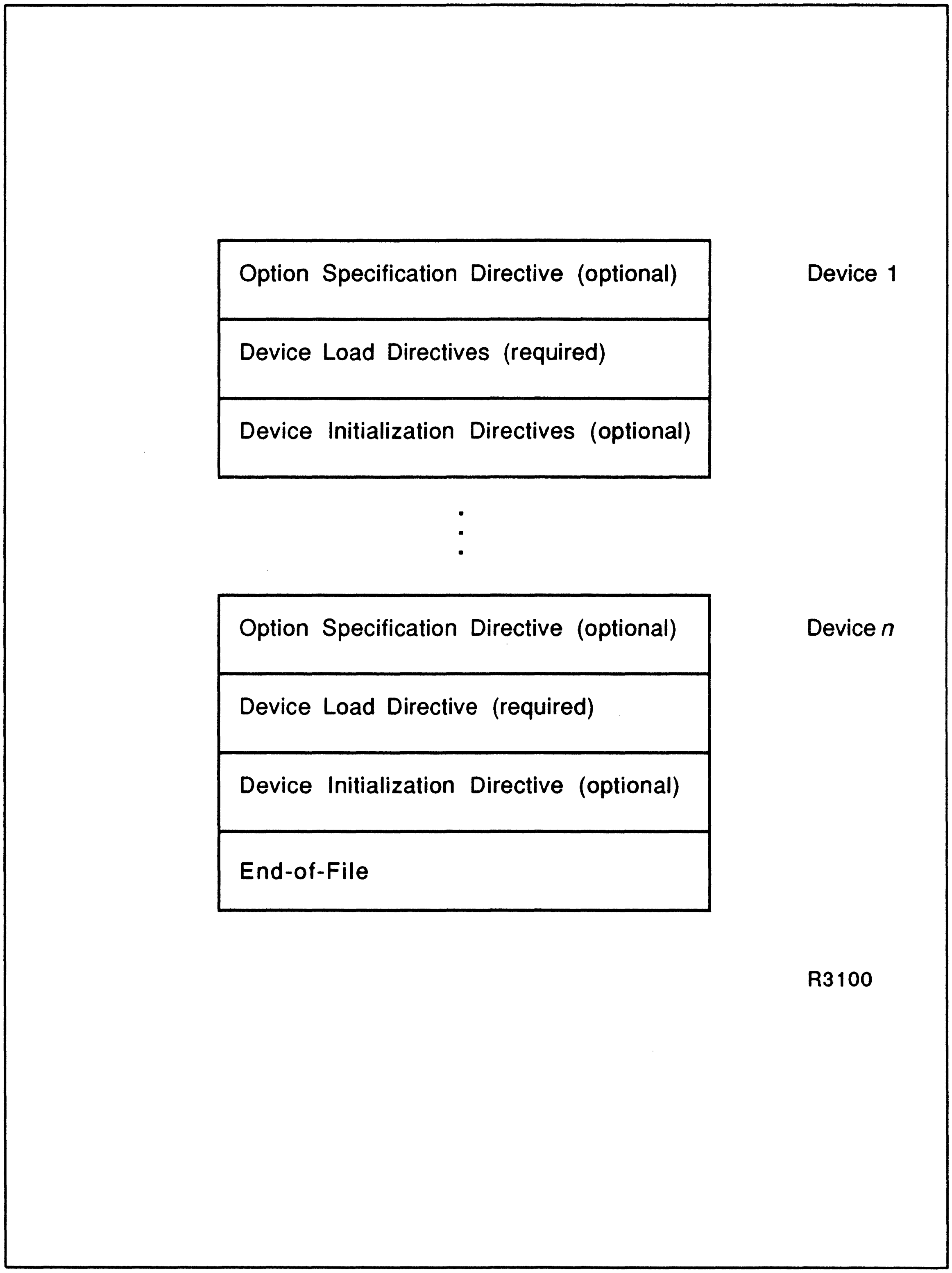

Figure 11-1
Initialization Directive File Format

11.3 Firmware File

Microcode is loaded by a single execute channel program call from a single buffer that is extracted from a firmware file on disk. When using TSM, the loading process is overridden by specifying option 9 before activating DEVINITL, in which case only the initialization process is performed.

As the microcode can be of any type, DEVINITL can load any WCS within the scope of its recognized record types.

The firmware file contains two types of records: data and control. All data record types are valid. Five control record types are valid:

- logical end-of-record
- header record
- set address-field length
- set checksum-field length
- terminate load record

These fields are further described in Table 11-1.

Table 11-1
Valid Control Record Types

Control Byte Value Decimal	Hex	Loader Control Record Description
0	0	Logical end-of-record – Remaining data in the current operating system record is ignored.
241	F1	Header record – The second byte of the record contains the number of data bytes in the record. The record can contain any file identification desired.
242	F2	Set address-field length – The next byte in the record defines the length of the address field in bytes for all subsequent loader data records.
243	F3	Set checksum-field length – The next byte in the record defines the length of the checksum field in bytes for all subsequent loader data records.
244-254	F4-FE	Reserved
255	FF	Terminate load record – This record indicates the end of the load. The address portion of this record specifies the initial transfer address for the control microcode previously read.

11.3.1 Loader Data Record Format

Loader data records define the data to be loaded into the device control memory (WCS). Each data record contains a byte count field, a microcode address field, a microcode data field, and a checksum field in the following format:

Record 0 *n*

Byte count See Note 1.	Firmware address. See Note 2.	Microcode data. See Note 3.	Checksum field. See Note 4.

Notes:

1. The byte count field defines the number of data bytes in the microcode data portion of the record. It also identifies the record as a loader data record as opposed to a loader control record. Valid byte counts are X'01' through X'F0'.
2. The firmware address field is a series of bytes defining the portion of the control memory where the associated data will be loaded. The default length of this field is 2 bytes; however, it can be changed with a Set Address-Field Length control record. The length specified remains in effect until another Set Address-Field Length control record is specified or end-of-file is encountered.
3. The microcode data field is a series of bytes containing the data to be loaded at the address specified, and in the format expected by the device. This format is the responsibility of the user; DEVINITL only checks record length and checksum.
4. The checksum field is a series of bytes containing the checksum value for the microcode data field. The default length of this field is 1 byte; however, it can be changed with a Set Checksum-Field Length control record. The length specified remains in effect until another Set Checksum-Field Length control record is specified or end-of-file is encountered.

11.4 Using DEVINITL

DEVINITL can be activated at IPL, in batch mode or in interactive mode, by TSM.

To activate DEVINITL at IPL time, use the SYSGEN ACTIVATE directive. Refer to the SYSGEN chapter in this manual. DEVINITL is activated immediately upon successful loading of the system image. When DEVINITL is activated at IPL time or in batch mode, milestone and error messages are displayed on the system console.

When DEVINITL is activated interactively, milestone and error messages are displayed on the terminal, and directives are passed to DEVINITL from the terminal. The first input required is a DEVICE directive. The initialization directive file is searched for the corresponding device entry.

If the device entry is found, any other valid directive can be entered. Any directive entered overrides the corresponding directive in that device's section. Any directive not entered defaults to that directive entry in the device's section. Terminal input only overrides initialization directive file directives; it does not permanently change them. A carriage return ends directive entry at the terminal.

If the device entry is not found, DEVINITL displays an appropriate message.

11.5 DEVINITL Directives

DEVINITL directives are summarized below, by command type. Each directive is described in detail in the remainder of this section, arranged alphabetically by directive name. Valid abbreviations are shown in bold type. In the initialization directive file, comments can be included in a directive line by specifying an exclamation point (!) anywhere on the line except within quoted ASCII strings. The remainder of the line is ignored.

All directives in an initialization directive file must be on separate lines.

When directives are entered through TSM, DEV_CNTRL is the only directive that must be on a separate line. All other directives can be continued across as many lines as necessary by placing a hyphen (-) as the last significant character on the line.

Option Specification Directive

OPTION	specifies whether parameter defaults for file matching and message displays are to be overridden for subsequent device entries in the initialization directive file. This directive is valid in the initialization directive file only.

Device Load Directives

DEVICE	specifies the device mnemonic, channel, and subchannel addresses for a device to be initialized and/or loaded. This directive is required for each device to be initialized and/or loaded. It must be the first command specified in the Device Load Directive section.
IDENT	specifies a header record ASCII string that must be matched if OPTION WCSMATCH=Y is specified. This directive is not required if OPTION WCSMATCH=N.
REREAD	specifies the number of read attempts to be performed on a firmware file if a read error occurs. The number specified is also used as the number of read attempts to be performed when the number of load attempts specified by a RETRY directive expires.
RETRY	specifies the number of load attempts to be performed on a device if a load error occurs.
WCS_FILE	specifies the pathname of a file containing microcode to be loaded onto a device. This directive is required if OPTION WCSMATCH=Y is specified or if microcode is to be loaded onto a device.

Device Initialization Directive

DEV_CNTRL	specifies device initialization is required.

11.5.1 DEV_CNTRL Directive

The DEV_CNTRL directive initializes a device. If this directive is not specified for a device, that device is not initialized. Special directives or data needed to perform the initialization can be specified.

When this directive is encountered, DEVINITL creates an input/output command doubleword (IOCD) using any data specified. The IOCD is then passed to the execute channel program for execution.

More than one DEV_CNTRL directive can be specified. However, only one DEV_CNTRL directive can be specified per input line.

Syntax

DEV_CNTRL [**CMD=***cc*] [*data* | '*string*']

[**CMD=***cc*] specifies the value of a valid hexadecimal IOCD directive byte. If not specified, default is X'FF' to initialize controller.

data is any valid hexadecimal data. Each byte of data must be separated by a blank or a comma. Data cannot be carried over one input line.

'*string*' is any ASCII data string enclosed in single quotes. Data cannot be carried over one input line. Its format is not checked.

If no parameters are specified, the device is initialized using system defaults.

Examples

The following example passes the ASCII string `STARTUP` to a device with a byte count of 7 with the initialization directive byte X'`F3`'.

```
DEV_CNTRL CMD=F3 'STARTUP'
```

The following example passes data `F081C2F7B177FF` to a device with a byte count of 7 with the default initialization command byte X'FF'.

```
DEV_ F0 81 C2 F7 B1 77 FF
```

11.5.2 DEVICE Directive

The DEVICE directive specifies a device that will be initialized and/or loaded. This is required in each device's device load directive section in the initialization directive file (see Figure 11-1), and must be the first directive in the section.

The device to be initialized and/or loaded must have been specified as a device at SYSGEN. If this directive is not specified or if an invalid device is specified, a fatal error occurs, ending the loading process using the initialization directive file.

Syntax

DEVICE=*mmccss*

mm	is a 2-character device mnemonic
cc	is a 2-character device channel address
ss	is a 2-character device subchannel address

Examples

The following example initializes and/or loads firmware onto the card reader (`CR`) on channel `20`, subchannel `08`.

```
DEVI=CR2008
```

11.5.3 IDENT Directive

The IDENT directive causes the header record of a firmware file to be searched for a specified ASCII string.

This directive is required if OPTION WCSMATCH=Y is specified. If a match is found, firmware loading proceeds. If a match is not found, an error is reported and DEVINITL proceeds to the next device entry in the initialization directive file.

Syntax

IDENT='*string*'

'*string*'	is any ASCII data string enclosed in single quotes. Data cannot be carried over one input line. Format is not checked.

Examples

The following example causes the header record of a firmware file to be searched for the string `FIRMWARE HEADER TY2.1`.

```
ID='FIRMWARE HEADER TY2.1'
```

11.5.4 OPTION Directive

The OPTION directive overrides DEVINITL defaults for header record searches and output message displays for subsequent device entries in the initialization directive file. This directive remains in effect until another OPTION directive is specified or the initialization directive file terminates.

This directive is optional and is valid in the initialization directive file only. Although both the WCSMATCH and MILESTONE parameters are optional, one of them must be specified if the OPTION directive is used. Both of them can be specified in one directive. Values yes (Y) and true (T) are synonymous. Values no (N) and false (F) are synonymous.

Syntax

OPTION [WCSMATCH={Y|N}] [MILESTONE={Y|N}]

[WCSMATCH={Y|N}]
specifies if subsequent header records in the firmware file must match an ASCII string which is specified by an IDENT directive. If Y and a match is not found, firmware loading is aborted. If N is specified, a match is not required and firmware loading proceeds. If not specified, the default is N.

[MILESTONE={Y|N}]
specifies whether milestone messages indicating successful operations are to be displayed or inhibited during subsequent device initialization and firmware loading. If Y is specified, messages are displayed. If N is specified, milestone messages are not displayed (but error messages are). If nothing is specified, default is Y.

Notes:

If this directive is specified, at least one of the parameters must be used.

Examples

The following example requires subsequent device entries in the initialization directive file to match the ASCII string specified with an IDENT directive. Milestone messages are not displayed.

```
OPTI WCSMATCH=Y MILESTONE=N
```

11.5.5 REREAD Directive

The REREAD directive specifies the number of read attempts performed on a firmware file when a read error occurs. When the specified number is exhausted, an error is reported.

The number specified by this directive is also used independently by the RETRY directive.

Syntax

REREAD=*num*

num is any decimal number 0 through 99999999. If not specified, the default is 1.

Examples

The following example specifies 3 read attempts are made on the firmware file if a read error occurs.

```
REREAD=3
```

11.5.6 RETRY Directive

The RETRY directive specifies the number of load attempts made on a device when a load error occurs. When the specified number is exhausted, DEVINITL automatically rewinds the firmware file and then attempts a read. The number of read attempts is the number specified in a REREAD directive.

When the RETRY and REREAD numbers are exhausted, an error is reported and DEVINITL proceeds to the next device entry in the initialization directive file.

Syntax

RETRY=*num*

num is a decimal number 0 through 99999999. If not specified, the default is 3.

Examples

The following example specifies that 5 load attempts are to be made on a device if a load error occurs. If loading is not successful after the fifth attempt, the firmware file automatically rewinds and read attempts are made, decrementing the REREAD directive specification.

```
RETRY=5
```

11.5.7 WCS_FILE Directive

The WCS_FILE directive specifies the pathname of a file containing microcode to be loaded onto a device.

This directive is required if OPTION WCSMATCH=Y is specified or if microcode is to be loaded onto a device. If an invalid pathname is specified or DEVINITL is denied access to the specified file, an abort occurs and an error is reported.

Syntax

WCS_FILE=*pathname*

pathname is the pathname of a file containing microcode

Examples

The following example loads the microcode file `WCS.1` located on the volume `SYSTEM` and directory `FIRMWARE` onto the specified device.

```
WCS_=@SYSTEM(FIRMWARE)WCS.1
```

11.6 Example Initialization Directive File

In the following initialization directive file, all directives are on separate lines.

```
OPTION WCSMATCH=Y MILESTONE=N       ! DEVICE 1
DEVICE=CR2008
ID='CARD READER FIRMWARE'
REREAD=0
RETRY=1
WCS_FILE=@SYSTEM(FIRMWARE)WCSCR
OPTION WCSMATCH=N                   ! DEVICE 2
DEVICE=TY7EB3
DEV_CNTRL CMD=F3 1B 1C
DEVICE=TY7EB4                       ! DEVICE 3
DEV_CNTRL CMD=F3 1E
DEVICE=TY7EB5                       ! DEVICE 4
DEV_CNTRL
OPTION WCSMATCH=N MILESTONE=Y       ! DEVICE 5
DEVICE=DC7EC0
REREAD=3
RETRY=5
WCS_FILE=@SYSTEM(FIRMWARE)WCSDISC
```

In the following example, directives are entered from TSM. Directives are continued across lines except for DEV_CNTRL, which is on a separate line. In this case, the hyphen signifies continuation of input for the same device.

```
TSM> DEVINITL DEVICE=CR2008 ID='CARD READER FIRMWARE'-
DEV> REREAD=0 RETRY=1-
DEV> DEV_CNTRL
```

11.7 Milestone Messages

The following message is displayed when a device is successfully initialized.

`DEVICE` *xxxxxx* `INITIALIZED`

The following message is displayed when firmware is successfully loaded onto a device.

`DEVICE` *xxxxxx* `HAS BEEN LOADED WITH THE FIRMWARE`
`CONTAINED IN WCS FILE` *xxxxxxxx*

11.8 Error Conditions and Messages

11.8.1 Initialization Directive File Errors

The following message is displayed when DEVINITL is activated by TSM and a corresponding device entry is not found in the initialization command file.

`NO DEFAULT DATA IS AVAILABLE FOR DEVICE` *xxxxxx*

The following message is displayed when action is not taken on a device entry within the initialization directive file.

`** WARNING ** NO ACTION HAS BEEN CARRIED OUT FOR DEVICE` *xxxxxx*

Action is not taken because one of the following occurred:

- option 9 is set
- a DEV_CNTRL command is not specified for the device entry
- a WCS_FILE directive is not specified for the device entry

The following message is displayed when the initialization directive file does not contain sufficient directives for processing a device before end-of-file detection.

`** ERROR ** A PREMATURE END-OF-FILE HAS BEEN DETECTED ON THE`
`CONTROL FILE`

The following message is displayed when a nondecimal number is specified with a REREAD directive.

`** ERROR ** INVALID REREAD PARAMETER SPECIFIED` *xxxxxxxx*

The following message is displayed when a nondecimal number is specified with a RETRY directive.

`** ERROR ** INVALID RETRY PARAMETER SPECIFIED` *xxxxxxxx*

11.8.2 Device Loading Errors

When a device cannot be successfully loaded, the following message is displayed along with the appropriate reason.

```
** ERROR ** UNABLE TO LOAD DEVICE xxxxxx FOR THE FOLLOWING REASON:
   DEVICE CHANNEL MUST BE HEXADECIMAL.
   DEVICE TYPE NOT CONFIGURED ON HOST SYSTEM.  FOUND xx
   UNABLE TO ALLOCATE DEVICE xxxxxx M.ASSN ERROR CODE:xx
   UNABLE TO ALLOCATE FILE xxxxxxxx M.ASSN ERROR CODE: xx
   WCS FILE xxxxxxxx IS INVALID - CONTAINS NO TERMINATION RECORD.
   WCS FILE xxxxxxxx IS INVALID - CONTAINS AN ERRONEOUS BYTE COUNT.
   OFFENDING RECORD BYTE COUNT: xx
   FIRMWARE ADDRESS: xxxx
   WCS FILE xxxxxxxx IS TOO LARGE FOR DEVINITL.
   OFFENDING RECORD BYTE COUNT: xx
   FIRMWARE ADDRESS: xxxx
```

Including microcode and data, maximum file size is 64K.

```
   WCS FILE xxxxxxxx CONTAINS INVALID COMMAND: HEX xx
   HARDWARE ERROR STATUS RECEIVED.  STATUS AFTER LAST RETRY:
   HEX xxxx
   UNABLE TO OPEN DEVICE xxxxxx DEVICE MAY NOT BE PRESENT
```

11.8.3 Directive Parsing Errors

The following message is displayed when an invalid pathname is specified with the WCS_FILE directive. The string *xxxxxxxx* contains up to 80 characters.

```
** ERROR ** PATHNAME SYNTAX ERROR: xxxxxxxx
```

The following message is displayed when OPTION WCSMATCH=Y and an IDENT directive have been specified, but a match is not found in the header record of the file being searched.

```
** WARNING ** INCORRECT FIRMWARE REVISION FOR DEVICE xxxxxx
   REQUESTED xxxxxxxxxxxxxxxx
   ACTUAL xxxxxxxxxxxxxxxx
```

The following message is displayed when the execute channel program call fails after the specified number of RETRY and REREAD attempts. The number output as the reason is the contents of the third word of the FCB.

```
DEVICE xxxxxx INITIALIZATION FAILED.
   REASON xxxxxxxx
```

The following message is displayed when a non-hexadecimal digit is found during IOCD creation at initialization time. The initialization line is ignored.

```
** ERROR ** INVALID HEXADECIMAL CHARACTER: x
```

The following message is displayed when the two least significant bits of the IOCD operation code directive specified with `DEV_CNTRL CMD=`*xx* are not set.

`** ERROR **` *xx* `IS INVALID IOCL COMMAND`

When a parse error occurs for a reason other than those described above, the following message is displayed along with the appropriate reason. Below this, the erroneous line is displayed with an exclamation point (!) below the item that caused the parse error.

`** PARSE ERROR **`

`AN EQUAL SIGN IS MISSING FOLLOWING A KEYWORD`

`PREMATURE END OF INPUT STRING`

A directive is truncated.

`DEVICE IDENTIFICATION IS MISSING`

`UNRECOGNIZABLE DIRECTIVE`

`A STRING IS MISSING FOLLOWING AN EQUAL SIGN`

`A QUOTED STRING IS IMPROPERLY TERMINATED`

An end quote is missing or attempt was made to carry a string past one input line.

`AN ASSUMED OPTION LINE CONTAINS NO OPTION DIRECTIVES`

Parameter WCSMATCH and/or MILESTONE must be specified with an OPTION command.

`A PREMATURE END-OF-STRING OCCURRED IN OPTION SPECIFICATION`

An OPTION directive cannot be carried past one input line.

`THE SPECIFIED OPTION IS UNDEFINED`

`RETRY/REREAD IS NOT A POSITIVE DECIMAL`

`INVALID OPTION SENSE SPECIFIED OPTIONS=Y,N,T,F, OR BLANK`

12 Alterable Control Store (ACS)

12.1 Introduction

The Alterable Control Store (ACS) Load and Display Utility performs two functions — LOADACS and DUMPACS. LOADACS writes firmware to the ACS or Writable Control Store (WCS). DUMPACS displays or compares firmware in Programmable Read Only Memory (PROM), ACS, and WCS. Through the SETCPU instruction, MPX-32 specifies whether the processor uses the PROM or ACS firmware.

With the exception of the CHECKSUM directive, the ACS Utility runs only on a CONCEPT 32/67 computer. The CHECKSUM directive can be used on any CONCEPT/32.

The numbers accepted and generated by LOADACS and DUMPACS are hexadecimal, except for decimal RM*xx* error codes.

12.2 LOADACS Directive File (M.ACS)

The LOADACS directives are in a file named M.ACS. M.ACS must be located on the system volume and the system directory. Use the Text Editor to create the file.

M.ACS consists of one or more groups of directives, each group for a particular processor. There can be more than one group of directives for any processor. The last directive in each group must be followed by a semicolon. Following is the format of M.ACS.

Group 1

COPY, LOAD, or PATCH Directive
VERIFY Directive (optional)
ENABLE Directive (optional)

.
.
.

Group *n*

COPY, LOAD, or PATCH Directive
VERIFY Directive (optional)
ENABLE Directive (optional)

12.3 Firmware File

The firmware file supplied must be restored as a system file using the Volume Manager.

The firmware file contains logical blocks of 256 bytes. The last byte of each logical block is the one's complement of the sum of all other bytes in the logical block. LOADACS uses this figure as a checksum during a LOAD.

If assigned, the firmware file must be assigned as unblocked.

12.3.1 CONCEPT 32/67 Usage

The CONCEPT 32/67 has two phases of processors, phase 1 and phase 2. The user must know which phase their system has so the M.ACS file can be properly constructed.

A properly constructed M.ACS directive loads or patches the proper firmware without user intervention regardless of the processor's phase. This maintains system portability from one phase processor to another.

The processor's phase is determined by the processor (CPU or IPU) PROM revision level. See the REVISION directive. If the last 8 digits of the revision number are 2282000*n*, where *n* is any number, the processor is a phase 1.

Any other numbers indicate a phase 2 processor.

12.3.2 Firmware File Record

The maximum record length is one logical block; therefore, the maximum record size is 251 bytes. The maximum number of 64-bit microwords per record is 31. The format of a record is:

0 7	8 23	24 *n*
Microword Count (Byte 0). See Note 1.	Start Address (Bytes 1 & 2). See Note 2.	Data (Bytes 3-250). See Note 3.

Notes:

1. Byte 0 of a record contains the microword count for the record. A count of 00 means the record is the last in a logical block.

   If the last record of a logical block ends in the last 3 bytes of the block, a new record is not started with a 00 microword count.

   An FF microword count indicates an end-of-file.

2. Bytes 1 and 2 of a record contain the starting address of the record's data. Valid address ranges are:

   0000 to 0FFF for ACS or PROM
   1000 to 1FFF for 4K x 64 WCS
   1000 to 2FFF for 8K x 64 WCS

3. Bytes 3-250 of a record contain the data to be loaded.

Microword FFD always contains the ACS or PROM firmware revision. The revision's format is:

C67*rnzzppppppzzzn*

C = CONCEPT/32
67 = 2-digit type of CONCEPT/32
r = firmware revision letter
n = firmware revision number
z = 0
p = 6-digit firmware part number

C67*rn*, with the variables filled, is the firmware source filename.

The following is an example of an ACS/PROM firmware revision.

C67A300322820003

The example's firmware source filename is C67A3. The firmware part number is 322820.

Microword FFE always contains the ACS or PROM data checksum. The checksum is the one's complement of the sum of the microwords. It is calculated by LOADACS during writes to ACS.

12.4 LOADACS

LOADACS is a subroutine in J.INIT that is executed during IPL and RESTART sequences. LOADACS receives commands from the M.ACS command file. If any of the following conditions are true, the firmware is not loaded and the load, patch, or copy message is not displayed:

- control switch 9 is set
- the machine is not a CONCEPT 32/67
- the sequence is not an IPL or RESTART
- M.ACS does not exist
- M.ACS is empty

LOADACS accepts different firmware and actions for the CPU and IPU. If the IPU is offline, a load request causes an error message to be displayed.

LOADACS runs in the processor it is loading. Before loading ACS or WCS, LOADACS executes the SETCPU instruction to enable transfer in the PROM mode. LOADACS blocks external interrupts in the CPU before loading, and unblocks them afterward. In the IPU, LOADACS inhibits context switching.

To load new firmware in ACS or WCS, use the LOAD directive. To load PROM firmware with modifications, use the COPY directive followed by a LOAD or PATCH directive. To modify ACS or WCS, use a LOAD or PATCH directive. PROM cannot be loaded or modified.

All LOADACS output is displayed on the operator console and the LOD device. A message is displayed after a successful load.

12.5 DUMPACS

DUMPACS is a subroutine in J.INIT. When J.INIT is called from TSM and option 1 is set, DUMPACS is the only subroutine called. DUMPACS then displays a DMP prompt for directives.

```
TSM> OPTION 1
TSM> J.INIT
DMP>
```

Depending on the directive, the results are written to logical file code UT or the LOD device. These logical file codes cannot be reassigned.

DUMPACS dumps the contents of ACS, WCS, and PROM, compare the contents of these storage areas with a firmware file, and calculate checksums to be used by the LOADACS PATCH directive. DUMPACS also displays the revision microword and the mode of a processor.

12.6 ACS Directives

ACS directives are summarized below, arranged by function type. Each directive is described in detail in the remainder of this section, arranged alphabetically by directive name.

LOADACS Directives

Directive	Description
COPY	copies contents of PROM to ACS
ENABLE	sets processor in ACS or PROM mode. This directive is not valid for WCS.
LOAD	loads firmware file into ACS or WCS
PATCH	loads data from M.ACS to ACS or WCS
VERIFY	compares ACS or WCS with the data that was written to ACS or WCS

DUMPACS Directives

Directive	Description
CHECKSUM	provides the checksum used by the PATCH directive
COMPARE	compares ACS, WCS, or PROM with a firmware file
DUMP	dumps specified memory contents to the LOD device
EXIT	exits DUMPACS and returns to TSM
MODE	displays the mode of a processor
REVISION	displays revision microword on LFC UT

12.6.1 CHECKSUM Directive

The CHECKSUM directive is used by DUMPACS to add values and compute the one's complement of the sum. The sum is then displayed on LFC UT.

When using LOADACS, the sum is the value entered in the optional SUM parameter for the PATCH directive.

The CHECKSUM directive can be continued across more than one input line by placing a hyphen (-) as the last significant character on the line to be continued. The hyphen must not be embedded in a value.

Syntax

CHECKSUM *value* [, *value* ...]

value is a value to be patched by LOADACS

Examples

The following example adds three values and computes the one's complement of the sum. The resulting value can be in the PATCH directive for the SUM parameter.

```
CHECKSUM 2433F55EC0024, 99FA5500, BB4302
```

12.6.2 COMPARE Directive

The COMPARE directive is used by DUMPACS to compare the contents of the specified ACS, WCS, or PROM with a firmware file. When the firmware file and memory do not match, the memory location's address and contents plus the file's contents are written to the LOD device. The firmware file must be formatted as described in section 12.3.2, Firmware File Record, of this chapter.

Syntax

COMPARE {**CPU** | **IPU**} {**ACS** | **PROM** | **WCS**} *pathname*

CPU specifies the CPU is the processor that is used

IPU specifies the IPU is the processor that is used

ACS specifies the ACS is read

PROM specifies the PROM is read

WCS specifies the WCS is read

pathname is the pathname of the firmware file. If the file does not exist, an error message is displayed.

Examples

The following example compares the contents of the `CPU's ACS` to the system file named `FIRM`.

```
COMPARE CPU ACS @SYSTEM(SYSTEM)FIRM
```

The following example compares the contents of the `IPU's WCS` to the user file named `FIRM`.

```
COMPARE IPU WCS FIRM
```

12.6.3 COPY Directive

The COPY directive is used by LOADACS to copy the contents of PROM to the CPU or IPU ACS.

Syntax

COPY {CPU|IPU}

CPU specifies the CPU ACS is written

IPU specifies the IPU ACS is written

12.6.4 DUMP Directive

The DUMP directive is used by DUMPACS to write the specified memory contents to the LOD device. If starting and ending addresses are not specified, the entire contents of the specified memory are written.

If an invalid address is specified, an error message is displayed on the terminal.

Syntax

DUMP {CPU | IPU} {ACS | PROM | WCS} [*startaddr*] [, *endaddr*]

CPU	specifies the CPU is the processor that is used
IPU	specifies the IPU is the processor that is used
ACS	specifies the ACS is read
PROM	specifies the PROM is read
WCS	specifies the WCS is read
startaddr	is the starting address of the dump
endaddr	is the ending address of the dump

Examples

The following example writes the entire contents of the `CPU's ACS` to the LOD device.

```
DUMP CPU ACS
```

12.6.5 ENABLE Directive

The ENABLE directive is used by LOADACS to set the processor which was just used for a COPY, LOAD, or PATCH, in the ACS or PROM mode. This directive is not valid for the WCS mode.

If the ACS mode is requested and a previous LOAD, COPY, PATCH, or VERIFY directive failed in ACS, the mode is not changed and an error message is written.

If the ENABLE operation is successful, the mode of the processor is reported to the console and the LOD device. When an ENABLE of IPU ACS causes the IPU to be marked offline due to incompatible firmware, the message IPU MARKED OFFLINE is displayed instead of the mode.

Syntax

ENABLE {ACS | PROM}

ACS	specifies the processor uses the ACS firmware
PROM	specifies the processor uses the PROM firmware

12.6.6 EXIT Directive

The EXIT directive exits DUMPACS and returns control to TSM.

Syntax

EXIT

12.6.7 LOAD Directive

The LOAD directive is used by LOADACS to load a firmware file into ACS or WCS. If an ACS load is not successful, the processor used remains in the PROM mode.

The LOAD directive can be continued across more than one line by placing a hyphen (-) as the last significant character on the line to be continued. The hyphen must not be embedded in a value and must directly precede the ACSMATCH or WCSMATCH keyword.

Syntax

LOAD {**CPU**|**IPU**} {**ACS**|**WCS**} **FILE**=*p1file* [,*p2file*]
[{**ACSMATCH**|**WCSMATCH**}=*p1rev* [*p2rev*]]

CPU specifies the CPU is the processor that is used

IPU specifies the IPU is the processor that is used

ACS specifies the firmware is loaded into ACS

WCS specifies the firmware is loaded into WCS

p1file specifies the 1- to 16-character name of the file containing the firmware loaded for a phase 1 processor. If *p2file* is not specified, *p1file* is loaded regardless of the processor type.

[,*p2file*] specifies the 1- to 16-character name of the file containing the firmware that is loaded for a phase 2 processor. When specified, either *p1file* or *p2file* is loaded depending on the processor's phase:

p1file	*p2file*	Processor Phase	File Loaded
present	not present	1	*p1file*
present	not present	2	*p1file*
present	present	1	*p1file*
present	present	2	*p2file*

ACSMATCH
compares the current ACS revision number to the number specified by *revnum*. Valid only for warm starts.

WCSMATCH
compares the current WCS revision number to the number specified by *revnum*. Valid only for warm starts.

p1rev is a 16-digit revision number. If *p1rev* and the current revision number do not match, a load does not take place and an error message is written to the LOD device. If a match occurs, the firmware is loaded regardless of the processor type.

[*p2rev*] is a 16-digit revision number. When specified, the firmware is loaded conditionally depending on the processor's phase:

p1rev	*p2rev*	Processor Phase	Action
present	present	1	revision number compared to *p1rev*. If matched, firmware is loaded. If not matched, no load occurs.
present	present	2	revision number compared to *p2rev*. If matched, firmware is loaded. If not matched, no load occurs.
present	not present	1	revision number compared to *p1rev*. If matched, firmware is loaded. If not matched, no load occurs.
present	not present	2	revision number compared to *p1rev*. If matched, firmware is loaded. If not matched, no load occurs.

Examples

The following example compares the `ACSMATCH` revision number to the current ACS revision number. If the numbers match, the contents of file `FIRM` are loaded into the `CPU`'s `ACS`.

```
LOAD CPU ACS FILE=FIRM ACSMATCH=C67A303228200003
```

The following example loads the contents of file `FIRM` into the `IPU`'s `WCS`.

```
LOAD IPU WCS FILE=FIRM
```

12.6.8 MODE Directive

The MODE directive is used by DUMPACS to display the mode of a processor on the device assigned to LFC UT.

Syntax

MODE {**CPU**|**IPU**}

CPU specifies the mode of the CPU

IPU specifies the mode of the IPU

12.6.9 PATCH Directive

The PATCH directive is used by LOADACS to modify the contents of ACS or WCS.

The PATCH directive can be continued across more than one input line by placing a hyphen (-) as the last significant character on the line to be continued. The hyphen must not be embedded in a value and must fall after the first value and before the SUM parameter.

Syntax

PATCH {CPU|IPU} [P1|P2] {ACS|WCS} LOC=*addr* **VALUE=***val* [,*val*...] **[SUM=***sum***]**

CPU specifies the CPU is the processor that is used

IPU specifies the IPU is the processor that is used

P1 specifies a phase 1 32/67 processor

P2 specifies a phase 2 32/67 processor

ACS specifies the ACS is patched

WCS specifies the WCS is patched

LOC=*addr*
: *addr* is the address in ACS or WCS loaded with the contents of M.ACS

VALUE=*val* [,*val*...]
: *val* is a value to be patched

[SUM=*sum***]**
: *sum* is a number to be compared with the one's complement of the sum of the values. If *sum* does not equal the one's complement of the sum of the values, an error message is written.

If the specified phase and the actual phase mismatch, a warning message is sent to the LOD and the rest of the command group is skipped. Failure to conditionally patch does not disable ACS.

Examples

The following example compares the one's complement of the sum of the values to the number in the `SUM` parameter. If the numbers match, the values are patched into the `CPU's ACS`.

```
PATCH CPU ACS LOC=E30 VALUE=4F55D234, 3534AD24-
,4536 SUM=FFFFFFFFB0753B3C
```

The following example patches location `1E30` in the `IPU's WCS` with the value `35F82`.

```
PATCH IPU WCS LOC=1E30 VALUE=35F82
```

In the following example, the first two directive groups are conditional patches depending on the 32/67 processor's phase. The third directive group is unconditionally patched, and the last directive group passes control to ACS.

```
PATCH CPU P1 ACS LOC=E30 VALUE=FEE, F1EF0EF00, FACE, FADE
 VERIFY;
PATCH CPU P2 ACS LOC=E3E VALUE=0,0, C0FFEE, D10DE
 VERIFY;
PATCH CPU ACS LOC=E29 VALUE=DEADDEAD, DEADDEAD
 VERIFY;
ENABLE ACS;
```

12.6.10 REVISION Directive

The REVISION directive is used by DUMPACS to display the revision microword on the device assigned to LFC UT.

Syntax

REVISION {CPU|IPU} {ACS|PROM|WCS}

CPU	specifies the CPU is the processor that is used
IPU	specifies the IPU is the processor that is used
ACS	specifies the ACS is read
PROM	specifies the PROM is read
WCS	specifies the WCS is read

12.6.11 VERIFY Directive

The VERIFY directive is used by LOADACS to compare the contents of ACS or WCS with an internal buffer. The buffer contains the data that was written to ACS or WCS as the result of the previous directive.

If the verification fails, an error message is written.

Syntax

VERIFY

12.7 Sample M.ACS File

The following is a sample M.ACS file.

```
COPY CPU;                                                    (Group 1)
PATCH CPU ACS LOC=E30 VALUE=4F55D234,35,34AD24-              (Group 2)
,4536 SUM=FFFFFFFFB0753B3C
VERIFY
ENABLE ACS;
LOAD IPU ACS FILE=IPUACS1 ACSMATCH=C67A300322820003          (Group 3)
ENABLE ACS;
```

When the sample file is executed, the phase 1 firmware in the CPU PROM is copied to the CPU ACS with the `COPY` directive. The semicolon at the end of the directive signals the end of a group in the M.ACS file.

The `PATCH` directive changes the contents of `CPU ACS` locations `E30` to E33 to the values `4F55D234`, `35`, `34AD24`, and `4536`. The one's complement of the sum of these numbers is compared to `FFFFFFFFB0753B3C`.

The `VERIFY` directive compares the CPU ACS with the contents in the internal buffer. The buffer contains the M.ACS and ACS contents used in the PATCH directive.

The `ENABLE` directive sets the CPU to the `ACS` mode. The semicolon at the end of the command signals the end of the second group of directives.

The `LOAD` directive reads the `IPU ACS` revision number. If the actual number does not match the revision number specified, LOADACS goes to the next group of directives. If the numbers match, the phase 1 firmware in system file `IPUACS1` is loaded into the `IPU ACS`.

The last `ENABLE` directive sets the IPU to the `ACS` mode if the LOAD was successful. The semicolon at the end of the directive signals the end of the third group of directives.

12.8 Error Conditions and Messages

12.8.1 LOADACS Error Conditions and Messages

The LOADACS error messages described in this section are written to the SLO device and the operator console.

The following message is displayed when the IPU is not online or SYSGENed in and cannot be loaded. The CPU can be loaded if necessary.

`J.INIT: CANNOT {COPY|LOAD|PATCH} IPU - IPU NOT ONLINE`

The following message is displayed when there is an unrecognizable directive in M.ACS. The directive is displayed and LOADACS goes to the next line of M.ACS.

`J.INIT: UNRECOGNIZED COMMAND IN M.ACS, IGNORED:`
directive

The following message is displayed when firmware cannot be loaded because of an invalid address in a firmware file. The address and the file name containing the address are written. LOADACS goes to the next group of directives.

`J.INIT: CANNOT {LOAD|PATCH} {ACS|WCS} FOR {CPU|IPU} - INVALID ADDRESS`
address `IN` *filename*

The following message is displayed when the machine type in the firmware's revision number is not C67. The machine type of the new firmware is displayed. LOADACS goes to the next group of directives.

`J.INIT: CANNOT {LOAD|PATCH} ACS FOR {CPU|IPU} - NEW FIRMWARE HAS M/C TYPE` *type*

One of the following messages is displayed when the ACSMATCH or WCSMATCH directive is in effect and the revision number of the current firmware and the number specified by ACSMATCH or WCSMATCH do not match. The current firmware's revision number is shown as PRESENT and the ACSMATCH or WCSMATCH revision number is shown as EXPECTED. LOADACS goes to the next group of directives.

`J.INIT: CANNOT LOAD {ACS|WCS} FOR {CPU|IPU} - PRESENT REV NUMBER NOT AS`
`EXPECTED FROM {ACSMATCH|WCSMATCH} COMMAND`
`PRESENT =` *revnumber*
`EXPECTED =` *revnumber*

The following message is displayed when a checksum error occurs during a LOAD, PATCH, or COPY. LOADACS goes to the next group of directives.

`J.INIT: CANNOT {LOAD|PATCH|COPY} {ACS|WCS} FOR {CPU|IPU} - CHECK SUM ERROR IN {`*filename*`|PROM}`

The following message is displayed when a verification fails. The address of the failure is displayed, plus the contents of the differing firmware. LOADACS goes to the next group of directives.

```
J.INIT: {COPY|LOAD|PATCH}  TO {CPU|IPU} FOR {ACS|WCS} FAILED VERIFICATION,
ADDRESS addr DIFFERS          {PROM|filename} = contents
                              {ACS|WCS} = contents
```

The following message is displayed when the firmware file cannot be assigned. A decimal error number (RM*xx*) is returned by H.REMM. LOADACS goes to the next group of directives.

```
J.INIT: CANNOT LOAD {ACS|WCS} FOR {CPU|IPU} - UNABLE TO ASSIGN filename
ERROR RMxx
```

The following message is displayed when an invalid value is found in a firmware file. The file's name, the invalid value, and the address of the value are displayed. LOADACS goes to the next group of directives.

```
J.INIT: CANNOT {LOAD|PATCH} {ACS|WCS} FOR {CPU|IPU} - INVALID VALUE
value IN  filename AT ADDRESS addr
```

The following message is displayed when an ENABLE directive is used in a group of WCS directives. The ENABLE directive is valid for ACS only. Remove the ENABLE directive.

```
J.INIT: ENABLE COMMAND IS INVALID WHEN WCS ACTION IS SPECIFIED
```

The following message is displayed when there is no WCS to load. LOADACS goes to the next group of directives.

```
J.INIT: CANNOT {LOAD|PATCH} {CPU|IPU}  WCS - NO WCS PRESENT
```

The following message is displayed if a previous action to a processor failed. J.INIT cannot enable the ACS since that firmware has not been properly loaded. LOADACS goes to the next group of directives.

```
J.INIT: CANNOT ENABLE {CPU|IPU} ACS - PREVIOUS ACTION TO THIS PROCESSOR FAILED
```

The following message is displayed if the processor is in the ROMSIM mode, i.e., writing to ACS results in a System Check Trap. LOADACS goes to the next group of directives.

```
J.INIT: CANNOT {COPY|LOAD|PATCH} {ACS|CPU} FOR {CPU|IPU} - {CPU|IPU} IS IN ROMSIM MODE
```

12.8.2 DUMPACS Error Conditions and Messages

The following DUMPACS error messages are written to the device assigned to LFC UT.

The following message displays when the last directive entered is invalid. Enter a valid directive at the DMP prompt.

```
INVALID COMMAND
```

The following message displays when the last directive entered contains an invalid processor mnemonic. The valid mnemonics are CPU and IPU.

```
INVALID PROCESSOR MNEMONIC
```

The following message displays when the last directive entered contains an invalid memory type. Valid memory types are PROM, ACS, and WCS.

```
INVALID MEMORY TYPE
```

The following message displays when the starting address specified in the DUMP directive is invalid. Re-enter the directive with a new starting address.

```
INVALID START ADDRESS
```

The following message displays when the ending address specified in the DUMP directive is invalid. Re-enter the directive with a new ending address.

```
INVALID END ADDRESS
```

The following message displays when the directives can be used only on a 32/67 machine.

```
THIS MACHINE IS NOT A 32/67 - CAN NOT DO IT
```

The following message displays when the IPU was not configured in the system at SYSGEN time.

```
IPU NOT CONFIGURED IN SYSTEM
```

The following message displays when the IPU is offline. IPU directives are not executed until the IPU is marked online.

```
IPU MARKED OFFLINE
```

The following message displays when DUMPACS cannot open LFC SLO to write required data. If the SLO default is used in this case, a system problem exists.

```
CAN NOT OPEN SLO
```

The following message displays when an incorrect address is found in the firmware file. Correct and reassemble the firmware file, then re-enter the directives.

BAD ADDRESS *addr* FOUND IN FIRMWARE FILE

The following message displays when the firmware file cannot be assigned. A decimal error number (RM*xx*) is returned by H.REMM.

UNABLE TO ASSIGN FILE - ERROR RM*xx*

The following message displays when a parameter in the last directive is invalid. Correct the parameter and re-enter the directive.

```
INVALID PARAMETER
```

The following message displays when the processor used in the directive does not have WCS.

```
THERE IS NO WCS ON THIS PROCESSOR
```

13 Volume Formatter (J.VFMT)

13.1 Introduction

The Volume Formatter (J.VFMT) formats volumes (disks) for use with MPX-32 and allows you to manage disk media online. With online media management directives, you can initialize media and perform media flaw editing.

When formatting volumes, J.VFMT FORMAT directive processing creates the following data structures on each volume:

Volume descriptor
Resource descriptor allocation bit map (DMAP)
DMAP descriptor
File space bit map (SMAP)
SMAP descriptor
System image descriptor
General allocatable resource descriptors
Allocatable resource descriptors map
Root directory
Root directory descriptor
DMAP deallocation file
DMAP deallocation file descriptor
SMAP deallocation file
SMAP deallocation file descriptor
Media deallocation file
Media deallocation file descriptor

13.2 General Description

J.VFMT operates in either of two environments: a fully functional MPX-32 system or a starter system generated by the System Distribution Tape (SDT) when SYSINIT activates J.VFMT from tape.

When IPL is performed from the SDT, file system support is not present in the form of a usable disk volume. To format a system volume, SYSINIT makes a static assignment indicating the system image is coming from tape, and a system image file is read from the SDT. Therefore, the IMAGE option for the FORMAT directive in J.VFMT must not be specified.

13.3 Logical File Code Assignments

Logical file codes required by J.VFMT are described in this section and summarized in Table 13-1.

13.3.1 Audit Trail (SLO)

As J.VFMT processes directives, it produces listed output for operations it performs. The LFC for the audit trail is SLO.

13.3.2 Directive Input (SYC)

Directive input is assigned to J.VFMT from logical file code SYC.

Table 13-1
J.VFMT Logical File Code Assignments

Input/Output Description	Logical File Code	Default assignments for J.VFMT
Audit trail	SLO	$ASSIGN SLO TO LFC=UT
Directive input	SYC	$ASSIGN SYC TO SYC

13.4 Using J.VFMT

J.VFMT formats unmounted volumes for use with MPX-32. An attempt to run J.VFMT on a mounted volume causes an abort.

Generally, only one J.VFMT directive verb can be processed for each FMT prompt. When the requested function completes, J.VFMT exits and returns to the calling task. For example,

```
TSM>$J.VFMT
FMT>FORMAT DEV=DM0802 VOL=EXAM MAXRE=3000 CON=Y
FMT>CON=N
DEVICE=DM0802 -- VOLUME FORMATTING SUCCESSFULLY COMPLETED
TSM>
```

An exception to this is for the INITIALIZE directive. After processing completes for this directive, J.VFMT prompts the user for an additional directive. This is useful (and required during an SDT activation of J.VFMT) to continue processing with the FORMAT directive.

When run in the batch mode, parameters must be specified in the first input string on the directive line.

In the interactive mode, J.VFMT prompts for more data after the first input string if:

- required parameters are not specified
- directive line contains a syntax error
- the Boolean flag for confirm is not turned off

If a syntax error occurs, J.VFMT displays an error message and the part of the directive line in error. The remainder of the directive line is ignored. However, directives and parameters in the directive line before the syntax error occurred are processed and need not be respecified.

To recover from certain errors in the interactive mode, J.VFMT requires specific information specified by error message output. In some cases, additional input is not accepted until requested.

13.4.1 Directive Syntax Rules

The directive line has three parts:

- Verb
- Parameters
- Options

Blank spaces are legal delimiters between directive line components.

13.4.1.1 Verb

The verb defines the action the utility is to take (such as FORMAT, REPLACE, EXIT) and determines the subset of valid options. The verb must be the first item on the directive line.

13.4.1.2 Parameters

Parameters are required keywords followed by an equal sign and a value. They are entered on the directive line after the verb. Parameters specify what is affected by the action of the directive. Parameters have three types of values:

- Boolean — Values are true (T), false (F), yes (Y), no (N), on (ON), and off (OFF). Values true, yes, and on are synonymous. Values false, no, and off are synonymous.
- Numeric — Values are either decimal or hexadecimal numbers. For example, radix can be stated using the radix indicators X and N, e.g., X'AB', N'18'.
- ASCII string — Values are expressed as strings containing alphanumeric characters.

13.4.1.3 Options

Options are optional keywords followed by an equal sign and a value. If specified, options are entered on the directive line after the parameters. Options are used as qualifiers (such as ACCESS=, CONFIRM=, IMAGE=). Where applicable, default values are provided. See the individual directive descriptions for defaults.

13.4.2 Directive Line Continuation

Directive lines can be continued across lines by placing a hyphen (-) as the last significant character on the line. This results in a prompt for more input when in the interactive mode, and another read when in batch mode or when reading a selected file. Each time a directive line is continued, the continuation character is replaced by a space. The only restriction is that single names, keywords, or an entire pathname must appear on the same line of input.

13.4.3 The CONFIRM Parameter

The CONFIRM parameter is included in the following Volume Formatter directives: EDITFMAP, FORMAT, INITIALIZE, NEWBOOT, and REPLACE. This parameter allows the user to review the specified values of all parameters and options associated with a directive. The user may then change these values before processing the directive, or begin processing without making any changes.

CONFIRM is only valid in interactive mode. In any other mode it is ignored.

When CONFIRM=Y (the default) is specified for a given directive, a list of that directive's options and parameters (and their specified values) is displayed as soon as the directive is entered. To change the values of any of these options or parameters, the user must type the option or parameter, an equals sign, and the new value. Changing an option or parameter requires the same syntax as specifying it as part of a directive. After the changes are entered, all options and parameters are displayed again. The directive may then be processed using the new values by entering `CONFIRM=N`.

Any number of options and parameters may be changed at one time. Changes can be made and displayed any number of times.

The user may also choose to exit Volume Formatter without processing the directive. This is done by typing `X` or `EXIT`.

13.5 Accessing the Volume Formatter

The Volume Formatter can be accessed from a batch file or TSM in the following ways:

```
$J.VFMT
$EXECUTE J.VFMT
$RUN J.VFMT
```
(valid from the system directory only)

When logical file code SYC is assigned to a terminal, the `FMT>` prompt is displayed.

13.6 Volume Formatter Directives

The following directives are supported by J.VFMT.

Directive	Description
COPY	copies a mounted volume to an unmounted volume
EDITFMAP	converts flaw definition information from one medium to another
EXIT	exits the Volume Formatter utility
FORMAT	software formats a disk
INITIALIZE	prepares a disk for volume formatting by writing new labels to its tracks and sectors, and builds a media descriptor (MD) track
NEWBOOT	writes a new bootstrap process to a formatted volume
REPLACE	writes a new default system image and bootstrap to a formatted volume

13.7 COPY Directive

The COPY directive copies a mounted volume to an unmounted medium. To do this, it formats the target medium with the characteristics of the source volume, then copies the RDs, directories, and files from the source to the target volume. The target medium must be the same disk type as the source. When using this directive with HSDP disks, the source and target disks must have the same track and sector configuration.

The COPY directive cannot be specified during SDT processing.

Syntax

COPY VOLUME=*volname* **TARGET=***mmccss* **[BRIEF={Y|N}] [DESTROY={Y|N}]**

VOLUME=*volname*

volname is the name of the source volume to copy

TARGET=*mmccss*

mmccss is the device type, channel address, and device subaddress of the device where the target medium is located. Valid device types are:

DM moving head or memory disk

FL floppy disk

DF fixed head disk

[BRIEF={Y|N}]

If N is specified, COPY writes to SLO a list of the directories and files copied and their size. If Y is specified, only the directories are written to SLO. However, if a file cannot be copied, its pathname and the applicable error code is written to SLO. Default is Y.

[DESTROY={Y|N}]

Y permits J.VFMT to overwrite a formatted target disk. The default is N. If the target disk is formatted but Y has not been specified, J.VFMT issues a warning message. In batch or command-file mode, J.VFMT then aborts to TSM. In interactive mode, J.VFMT follows the warning message with the prompt `CONTINUE - Y/N?`. If the user chooses `Y`, the disk is overwritten. Otherwise, J.VFMT aborts to TSM.

Notes:

1. J.VFMT does not copy files in the following cases:
   - source file cannot be allocated for implicit shared read access with no previous writers on the file
   - target volume lacks enough contiguous file space for the source file or for the directory file
   - source file is in a directory created after the disk copy begins
   - source file or partition is created after processing of its directory begins
   - source file or partition is deleted before its directory entry is processed
2. Error messages display when:
   - a file or directory cannot be copied
   - an RID is changed
   - a file's start address is not maintained at the same start address
   - the bootability of a volume is not maintained
3. General SLO output contains:
   - a list of directories copied
   - a snapshot of the free space on the source volume when COPY is initiated
   - a snapshot of the free space on the target volume when COPY concludes, including the amount of contiguous free space
   - disk bus performance information. This includes the number of blocks transferred, the elapsed time of the copy, and a bytes per second rate with a bus efficiency rating.
   - if BRIEF=N is specified, a list of the files copied and their size

13.8 EDITFMAP Directive

The EDITFMAP directive converts flaw definition information from one medium to another. This directive is used for online disk media management. For more information about managing disk media online, refer to the Online Disk Media Management section in this chapter.

The EDITFMAP directive executes in either read-only mode or no I/O mode.

- Use read-only mode when the disk is present and has flaw information available for read-only access. To specify this mode include both the DISC and DEVICE options with this directive.
- Use no I/O mode when the disk cannot or need not be read. To specify this mode include the DISC option and omit the DEVICE option with this directive.

The EDITFMAP directive cannot be specified during SDT processing.

Syntax

EDITFMAP **DIS**C=*dcode* [**CON**FIRM={Y|N}] [**DEV**ICE=*mmccss*] [**DUM**P=(*ddccss,id1*)]
[**FMA**P={Y|N}] [**INC**LUDE=*mapfile*] [**LOA**D=(*ddccss,id2*)]

DISC=*dcode*

dcode is the disk storage device type code. Refer to Note 1 in the INITIALIZE directive section for a list of disk type codes.

[**CON**FIRM={Y|N}]

specifies whether confirmation is required before processing begins. If Y is specified, prompts are issued which allow the user to modify options and parameters, or start processing. If N is specified, no prompt is issued. The default is Y. This option is valid only in interactive mode; it is ignored in any other mode.

[**DEV**ICE=*mmccss*]

mmccss is the device type, channel address, and device subaddress of the device with the media flaw information to edit. Valid device types are:

DM moving head or memory disk

FL floppy disk

DF fixed head disk

[DUMP=(*ddccss,id1*)]
specifies writing media flaw information in dumped media flaw format to a device and assigns an ID to the output. See note 1.

ddccss is the device mnemonic, channel address and device subaddress. The mnemonic can be any magnetic tape mnemonic (M7, M9, MT) or floppy disk (FL).

id1 is an identifier (1 to 20 ASCII characters) to record on the external medium with the dumped media flaw data. Successive single quote marks indicate that *id1* contains all blanks.

[FMAP={Y|N}]
Y writes the media flaw map to SLO. The default is N.

[INCLUDE=*mapfile*]
mapfile is the pathname of a file containing a media flaw map to include in the new media flaw data being generated. Refer to section 13.16.6 for a description of the format for a media flaw map entry.

[LOAD=(*ddccss,id2*)]
specifies read of media flaw information in dumped media flaw format from a device. See note 1.

ddccss is the device mnemonic, channel address and device subaddress. The mnemonic can be any magnetic tape mnemonic (M7, M9, MT) or floppy disk (FL).

id2 is an identifier (1 to 20 ASCII characters) to compare to the ID recorded on the external medium with the dumped media flaw data being read. Successive single quote marks indicate that verification of *id2* is not required.

Notes:

1. The LOAD and DUMP identifiers are incompatible with early releases of the UDP/DP II MVP that limit identifiers to 4 characters.

Examples

The following example converts dumped media flaw data from tape drive `M91000` to SLO.

```
EDI DIS=DC0600 CON=N FMAP=Y LOA=(M91000,FLAWS.D1)
```

The following example converts flaw definition information from the text file `FILE1` to SLO.

```
EDI DIS=DP0080 FMA=Y INC=@RT0(SYSTEM)FILE1
```

13.9 EXIT Directive

The EXIT directive exits J.VFMT and returns control to TSM.

Syntax

{**EXIT** | **X**}

13.10 FORMAT Directive

The FORMAT directive software formats a disk. This directive does not check for existing data on the disk, but does check for existing volume structures.

For all disks, except floppy and memory disks, the disk media must have been initialized prior to using the FORMAT directive. This may be accomplished by:

- using the J.VFMT INITIALIZE directive (MFP SCSI, HSDP, or UDP/DP II controllers)
- running the Level II Diagnostic Media Verification program (HSDP or UDP/DP II controllers)
- running the Level II Diagnostic program (MFP SCSI)

Syntax

FORMAT **DEV**ICE=*mmccss* **VOL**UME=*volname* [**ACC**ESS=*usertype*([**R**EAD] [**D**ELENT] [**A**DD] [**T**RAVERSE])]... [**ALO**C=*blocks*] [**BOO**TFILE=**OLD** | **NEW** | *bstrap*}] [**CON**FIRM={**Y** | **N**}] [**DES**TROY={**Y** | **N**}] [**DET**ACH={**Y** | **N**}] [**ECC**={**Y** | **N**}] [**IMA**GE=*image*] [**ISI**ZE=*blocks*] [**MAXA**U=*units*] [**MAXRE**S=*rds*] [**MAXRO**OT=*entries*] [**OWN**ER=*ownername*] [**PRO**JECTGROUP=*proj*]

DEVICE=*mmccss*
: *mmccss* is the device type, channel address, and device subaddress of the disk to be formatted. Valid device types are:

DM	moving head or memory disk
FL	floppy disk
DF	fixed head disk

VOLUME=*volname*
: *volname* is a user-specified name for the volume being formatted. This name can be 1 to 16 characters.

[ACCESS=*usertype*([READ] [DELENT] [ADD] [TRAVERSE])]...
specifies the access attributes for the root directory of the volume. Empty parentheses, such as `ACCESS=OT( )` indicate no access possible. Repeat this option for each *usertype* desired. If not specified, owner has complete access, project group has all but delete access, and others have read only and traverse access.

usertype is one of the following types of users:

- **OWNER** specifies what access rights the owner has to the root directory
- **PROJECTGROUP** specifies what access rights the project group has to the root directory
- **OTHERS** specifies what access rights others have to the root directory

READ permits *usertype* to read the root directory

DELENT permits *usertype* to delete entries from the root directory

ADD permits *usertype* to add entries to the root directory

TRAVERSE permits *usertype* to traverse the root directory

[ALOC=*blocks*]
blocks is the number of blocks per allocation unit on the disk being formatted. The maximum value for *blocks* is the number of sectors per track. If omitted or set to 0, J.VFMT uses a system default based on the disk's capacity. For information on allocation units, refer to Table 13-2.

[BOOTFILE={OLD | NEW | *bstrap*}]
specifies the bootstrap to be installed to the disk. If this option is omitted, J.VFMT installs the appropriate standard bootstrap.

OLD specifies the old standard bootstrap. The old standard bootstrap is designed for images that require no more than 5 IOCDs to load.

NEW specifies the new standard bootstrap. The new standard bootstrap is designed for images that require 6 or more IOCDs to load.

bstrap is the pathname of a file containing a nonstandard bootstrap. This bootstrap may follow either the old or new style. This file must be in absolute load module format. Its maximum length is seven 192-word blocks including the load module preamble. **Note:** Do not specify *bstrap* when J.VFMT is activated from an SDT.

Not all MFP SCSI interfaces support new-style bootstraps. If you specify `NEW`, or a pathname containing a nonstandard bootstrap designed for images requiring 6 or more IOCDs, when formatting an MFP SCSI interface which does not support the new-style bootstrap, J.VFMT aborts to TSM.

For more information on the old and new standard bootstraps, see Chapter 5, section 5.2, of this volume.

[CONFIRM={Y|N}]
specifies whether confirmation is required before processing begins. If Y is specified, prompts are issued which allow the user to modify options and parameters, or start processing. If N is specified, no prompt is issued. The default is Y. This option is valid only in interactive mode; it is ignored in any other mode.

[DESTROY={Y|N}]
Y permits J.VFMT to overwrite existing volume structures on the disk to be formatted. The default is N. If the disk to be formatted contains existing volume structures but Y has not been specified, J.VFMT issues a warning message. In batch or command-file mode, J.VFMT then aborts to TSM. In interactive mode, J.VFMT follows the warning message with the prompt `CONTINUE - Y/N?`. If the user chooses `Y`, the disk is overwritten. Otherwise, J.VFMT aborts to TSM.

[DETACH={Y|N}]
specifies whether J.VFMT should run as an independent task. If Y is specified, J.VFMT displays the current parameters, then detaches from the terminal, and returns control to TSM. Formatting continues as usual. When it completes, a message displays on the terminal. This option is valid only in interactive mode.

[ECC={Y|N}]
specifies whether disk sectors with correctable data errors are available for users to allocate. If Y is specified, users can allocate these sectors. If N is specified, J.VFMT marks these sectors as allocated (bad block deallocation). The default is N.

[IMAGE=*image*]
image is the pathname of a file containing a system image. If the IMAGE option is not specified and J.VFMT is disk activated, the target volume will not be bootable. **Note:** Do not specify this option when J.VFMT is activated from an SDT.

[ISIZE=*blocks*]
blocks is the number of blocks allocated for the system image. The default is 600.

[MAXAU=*units*]
units is the number of allocation units on the disk. This number's default value depends on the ALOC=*blocks* option or the default granularity if *blocks* is not specified. If specified, *units* must be less than the number of allocation units physically available.

[MAXRES=*rds*]
rds is the number of resource descriptors (RD) to reserve disk space for. One RD is required for every file defined on the volume. The default for memory disks or floppy disks is 70. All other devices default to 1000.

[MAXROOT=*entries*]
entries is the number of entries in the root directory to reserve disk space for. This number can be modified to accommodate various hashing algorithms. The default is 100.

[OWNER=*ownername*]

ownername is the owner name to associate with the volume. This name is recorded in the volume descriptor. If omitted, J.VFMT uses the owner name associated with the J.VFMT task.

[PROJECTGROUP=*proj*]

proj is the project group name to associate with the volume. This name is recorded in the volume descriptor. If omitted, J.VFMT uses the project group name associated with the J.VFMT task.

Table 13-2
Allocation Units

Capacity	Drive type	Default sectors per unit
N/A	floppy moving head	1
N/A	memory disk	1
5 MB	fixed head	1
32 MB	cartridge module	2
40 MB	moving head	2
64 MB	cartridge module	2
80 MB	moving head	2
96 MB	cartridge module	2
160 MB	moving head	4
300 MB	moving head	4
337 MB	moving head	5
340 MB	moving head	4
447 MB	moving head	5
500 MB	moving head	5
600 MB	moving head	10
687 MB	moving head	9
700 MB	moving head	7
800 MB	moving head	9
850 MB	moving head	9
858 MB	moving head	9
1230 MB	moving head	11

13.11 INITIALIZE Directive

The INITIALIZE directive prepares a disk for volume formatting by writing new track and sector labels, and building the media descriptor track. This directive is used for online disk media management. For more information about managing disk media online, refer to the Online Disk Media Management section in this chapter.

Because this directive overwrites the disk, for HSDP disk media save any existing vendor flaw information to a device before initialization or by specifying the DUMP option with this directive.

This directive does not initialize the two cylinders reserved for diagnostic use or the cylinder reserved for hardware vendor use.

Executing the INITIALIZE directive can be a lengthy process. Depending on the disk capacity, the directive can take in the order of 30 to 45 minutes to complete. In the interactive mode, J.VFMT writes status information to the terminal, allowing the user to follow its progress. For MFP SCSI media, no status information is generated, so the user must allow an appropriate time for completion.

With the INITIALIZE directive you can:

- read media flaw information from an ASCII text file by specifying the INCLUDE option
- read dumped flaw information from a device by specifying the LOAD option
- check for additional media flaws on a disk by specifying PATTERN=Y
- initialize a disk for dual port use by specifying DUAL=Y
- write new media flaw information in dumped media flaw format with the DUMP option
- write a media flaw map to SLO by specifying FMAP=Y
- perform sector substitution by specifying SSUB=Y. (HSDP disk only)

Syntax

INITIALIZE **DEV**ICE=*mmccss* [**CON**FIRM={Y|N}] [**DES**TROY={Y|N}] [**DIS**C=*dcode*]
[**DU**AL={Y|N}] [**DUM**P=(*ddccss,id1*)] [**FMA**P={Y|N}] [**INC**LUDE=*mapfile*]
[**LOA**D =(*ddccss,id2*)] [**PAT**TERN={Y|N}] [**SSU**B={Y|N}]

DEVICE=*mmccss*
: *mmccss* is the device type, channel address, and device subaddress of the disk to be initialized. Valid device types are:

DM moving head or memory disk

FL floppy disk

DF fixed head disk

[**CON**FIRM={Y|N}]
: specifies whether confirmation is required before processing begins. If Y is specified, prompts are issued which allow the user to modify options and parameters, or start processing. If N is specified, no prompt is issued. The default is Y. This option is valid only in interactive mode; it is ignored in any other mode.

[**DES**TROY={Y|N}]
: Y permits J.VFMT to overwrite a formatted disk. The default is N. If the disk to be initialized is formatted but Y has not been specified, J.VFMT issues a warning message. In batch or command-file mode, J.VFMT then aborts to TSM. In interactive mode, J.VFMT follows the warning message with the prompt `CONTINUE - Y/N?`. If the user chooses `Y`, the disk is overwritten. Otherwise, J.VFMT aborts to TSM.

[DISC=*dcode*]
dcode is the disk device type code. Refer to Note 1 for a list of disk type codes. **Note:** This option is required when the target disk is an HSDP media that has been sector substituted.

[DUAL={Y|N}]
Y writes track and sector labels with the dual port option bit set in the DATR field of each label. The default is N.

[DUMP=(*ddccss*,*id1*)]
specifies writing media flaw information in dumped media flaw format to a device and assigns an ID to the output. See note 2.

ddccss is the device mnemonic, channel address and device subaddress. The mnemonic can be any magnetic tape mnemonic (M7, M9, MT) or floppy disk (FL).

id1 is an identifier (1 to 20 ASCII characters) to record on the external medium with the dumped media flaw data. Successive single quote marks indicate that *id1* contains all blanks.

[FMAP={Y|N}]
Y writes a media flaw map to SLO after the disk is initialized. The default is N.

[INCLUDE=*mapfile*]
mapfile is the pathname of a file containing a media flaw map to include in the new media flaw data being generated. Refer to section 13.16.6 for a description of the format for a media flaw entry.

[LOAD=(*ddccss*,*id2*)]
specifies read of media flaw information in dumped media flaw format from a device. See note 2.

ddccss is the device mnemonic, channel address and device subaddress. The mnemonic can be any magnetic tape mnemonic (M7, M9, MT) or floppy disk (FL).

id2 is an identifier (1 to 20 ASCII characters) to compare to the ID recorded on the external medium with the dumped media flaw data being read. Successive single quote marks indicate that verification of *id2* is not required.

[PATTERN={Y|N}]
Y tests the disk for flaws by writing a worst-case pattern to the disk, then reading and comparing it. Any errors detected are added to the media flaw information. The default is N.

[SSUB={Y|N}]
Y substitutes the spare sector on each of the disk's tracks for a single flawed sector on that track. This option is valid only for disks with HSDP disk interfaces. It is ignored when intializing disks with any other type of disk interface. The default is N.

Notes:

1. The disk device type codes valid for the INITIALIZE and EDITFMAP directives are as follows.

Disk Type Code	Controller	Model Number	Description
DP0080	HSDP	8138	CDC 80 MB RSD
DP0337	HSDP	8887	Fujitsu 337 MB Winchester
DP0474	HSDP	8884	Fujitsu 474 MB Winchester
DP0500	HSDP	8812	CDC 500 MB EMD
DP0689	HSDP	8889	Fujitsu 689 MB Winchester
DP0800	HSDP	8881	NEC 800 MB Winchester
DP0850	HSDP	8813	CDC 850 MB EMD
DP0858	HSDP	8888	CDC 858 MB XMD
DP1230	HSDP	8814	CDC 1.23 GB EMD
DC0080	UDP/DP II	var.	CDC 80 MB disk
DC0160	UDP/DP II	8127	CDC 160 MB fixed media
DC0300	UDP/DP II	9346	CDC 300 MB removable
DC0340	UDP/DP II	8858	CDC 340 MB Winchester
DC0600	UDP/DP II	8155	CDC 675 MB fixed media
SD0150	MFP SCSI	8820	CDC 150 MB Wren III SCSI
SD0300	MFP SCSI	8828	CDC 300 MB Wren IV SCSI
SD0700	MFP SCSI	8833	CDC 700 MB Wren V SCSI

   For UDP/DP II, the SYSGEN disk type codes MH*xxx* can be used instead of DC0*xxx*. This enables compatibility with existing disk type codes defined in the SYSGEN file.

2. The LOAD and DUMP *ids* are incompatible with early releases of the UDP/DP II Media Verification Programs that limit the *id* field to 4 characters.

Examples

The following example initializes the disk on device `DM0800` for multiport use. Before initialization, it writes the disk's existing media flaw information to device `FL7EF0` and assigns it an ID of blanks. After initializating the disk, J.VFMT writes a media flaw map to SLO.

In non-interactive mode, J.VFMT aborts if the target device, `DM0800`, is found to be software formatted. In interactive mode, J.VFMT prompts the user to continue if the the target device is formatted.

```
INI DEV=DM0800 DUAL=Y DUMP=(FL7EF0,'') FMAP=Y
```

The following example initializes the disk type DP0500 on device DM0C00. It includes the media flaw map in MAP500 file and media flaw information from device M91000 in the new Media Descriptor (MD) track. Any existing structures on the disk are overwritten.

```
INIT DEV=DM0C00 DES=Y DISC=DP0500 INC=MAP500-
LOA=(MT91000,RTSW)
```

13.12 NEWBOOT Directive

The NEWBOOT directive replaces the bootstrap on a previously formatted volume. For this volume to be bootable, you must have specified the IMAGE option when formatting the disk with J.VFMT. See the FORMAT directive. The NEWBOOT directive is invalid when J.VFMT is activated from an SDT.

Syntax

NEWBOOT **DEV**ICE=*mmccss* **VOL**UME=*volname* [**BOO**TFILE={**OLD** | **NEW** | *bstrap*}]
[**CON**FIRM={**Y** | **N**}]

DEVICE=*mmccss*

mmccss is the device type, channel address, and device subaddress of the disk whose bootstrap program is being replaced. Valid device types are:

DM moving head or memory disk

FL floppy disk

DF fixed head disk

VOLUME=*volname*

volname is the name of the target volume.

[**BOO**TFILE={**OLD** | **NEW** | *bstrap*}]

specifies the bootstrap to be installed to the disk. If this option is omitted, J.VFMT installs the appropriate standard bootstrap.

OLD specifies the old standard bootstrap. The old standard bootstrap is designed for images that require no more than 5 IOCDs to load.

NEW specifies the new standard bootstrap. The new standard bootstrap is designed for images that require 6 or more IOCDs to load.

bstrap is the pathname of a file containing a nonstandard bootstrap. This bootstrap may follow either the old or new style. This file must be in absolute load module format. Its maximum length is seven 192-word blocks including the load module preamble.

Not all MFP SCSI interfaces support new-style bootstraps. If you specify NEW, or a pathname containing a nonstandard bootstrap designed for images requiring 6 or more IOCDs, for an MFP SCSI interface which does not support the new-style bootstrap, J.VFMT aborts to TSM.

For more information on the old and new standard bootstraps, see Chapter 5, section 5.2, of this volume.

[CONFIRM={Y|N}]
specifies whether confirmation is required before processing begins. If Y is specified, prompts are issued which allow the user to modify options and parameters, or start processing. If N is specified, no prompt is issued. The default is Y. This option is valid only in interactive mode; it is ignored in any other mode.

13.13 REPLACE Directive

The REPLACE directive writes a new default system image and bootstrap to a formatted volume. Use this directive to modify system image data in the volume descriptor. The volume can already contain a default system image.

Syntax

REPLACE DEVICE=*mmccss* **IMAGE=***image* **VOLUME=***volname*
[BOOTFILE={OLD|NEW|*bstrap***}] [CONFIRM={Y|N}]**

DEVICE=*mmccss*
mmccss is the device type, channel address, and device subaddress to write a new system image and bootstrap. Valid device types are:

DM moving head or memory disk

FL floppy disk

DF fixed head disk

IMAGE=*image*
image is the pathname of a file containing a system image.

VOLUME=*volname*
volname is the name of the target volume.

[BOOTFILE={OLD|NEW|*bstrap***}]**
specifies the bootstrap to be installed to the disk. If this option is omitted, J.VFMT installs the appropriate standard bootstrap.

OLD specifies the old standard bootstrap. The old standard bootstrap is designed for images that require no more than 5 IOCDs to load.

NEW specifies the new standard bootstrap. The new standard bootstrap is designed for images that require 6 or more IOCDs to load.

bstrap is the pathname of a file containing a nonstandard bootstrap. This bootstrap may follow either the old or new style. This file must be in absolute load module format. Its maximum length is seven 192-word blocks including the load module preamble. **Note:** Do not specify *bstrap* when J.VFMT is activated from an SDT.

Not all MFP SCSI interfaces support new-style bootstraps. If you specify NEW, or a pathname containing a nonstandard bootstrap designed for images requiring 6 or more IOCDs, when using REPLACE with an MFP SCSI interface which does not support the new-style bootstrap, J.VFMT aborts to TSM.

For more information on the old and new standard bootstraps, see Chapter 5, section 5.2, of this volume.

[CONFIRM={Y|N}]
specifies whether confirmation is required before processing begins. If Y is specified, prompts are issued which allow the user to modify options and parameters, or start processing. If N is specified, no prompt is issued. The default is Y. This option is valid only in interactive mode; it is ignored in any other mode.

13.14 J.VFMT Errors

The following error messages are issued by J.VFMT:

Code	Description
VF01	ERROR HAS OCCURRED. SEE SLO FILE FOR EXPLANATION.
VF02	OPEN FAILURE ON AUDIT TRAIL DEVICE/FILE
VF03	EOF/EOM ON AUDIT TRAIL DEVICE/FILE
VF04	I/O ERROR ON AUDIT TRAIL DEVICE/FILE

13.15 Examples

The following example formats a non-bootable disk using the name MASTER. It is located on device DM0800, and space for 4000 resource descriptors is reserved.

```
FMT>FORMAT DEV=DM0800 VOL=MASTER MAXRE=4000
```

The following example writes the nonstandard bootstrap specified by the BOOTFILE parameter to volume NEW located on device DM0802. All parameters will be displayed. Confirmation will have to be reset to false (i.e., CON=N) to continue the formatting process.

```
FMT>NEWBOOT DEV=DM0802 VOL=NEW-
FMT_BOOTFILE=@SYSTEM(USER1)BOOTSTRAP-
FMT_CON=Y
```

The following example changes the default system image on volume MASTER4 located on device DM0804 to the system image defined in file BOOT located on volume SYSTEM in directory USER1. The appropriate standard bootstrap is also written to the disk.

```
FMT>REPLACE DEV=DM0804 VOL=MASTER4-
FMT_IMA=@SYSTEM(USER1)BOOT
```

The following example formats a disk using the name MASTER. It is located on device DM0800, the owner has complete access to the volume, the project group has read and add access to the volume, and other users have read access to the volume.

```
FMT>FORMAT DEV=DM0800 VOL=MASTER-
FMT_ACC=OW(R A D T)-
FMT_ACC=PR(R A )-
FMT_ACC=OT(R)
```

13.16 Online Disk Media Management

To manage disk media online, use the J.VFMT INITIALIZE and EDITFMAP directives. These directives provide media initialization and media flaw editing for the following disk interfaces:

- universal disk processor (UDP)
- disk processor II (DP II)
- high speed disk processor (HSDP)

Media management using J.VFMT is generally equivalent to the CONCEPT/32 Diagnostic Facility Level II Media Verification Program (MVP) for these disk interfaces.

For small computer system interface (SCSI) disks, J.VFMT provides limited media management capabilities. Refer to the Managing SCSI Disks section later in this chapter for more information.

The following sections define media management terminology and describe how to initialize disks and manage media flaw information.

13.16.1 Media Management Terminology

This section defines the terms used in following discussions of media management directives and procedures. It also notes where HSDP or UDP/DP II use different terms for similar structures.

Media flaw is a physical defect on a disk. The term *media flaw* is interchangeable with the term *media defect*. This chapter uses *flaw*, except where *defect* is already part of a data structure name.

Vendor flaw information describes disk media flaws detected by the vendor during factory testing. This information is usually recorded on each track of the disk in a structure called a vendor label. HSDP can read vendor flaw information with the Read Vendor Label command. UDP/DP II do not have this capability.

Media flaw information describes a media flaw in its raw or unformatted track location. This raw definition is used internally by J.VFMT when processing flaws and most precisely describes the actual location of the flaw on the disk. J.VFMT converts all flaw formats to this format for processing and converts them to the requested output format after processing.

Media deallocation data lists the disk sectors that have media flaws or are otherwise reserved. This data (or data lists) is recorded on the media descriptor track in a prescribed format and is used in bad block deallocation. HSDP refers to this information as the *media flaw map* and UDP/DP II uses the term *media descriptor.*

Media descriptor (MD) track contains media deallocation data. It is built by the INITIALIZE directive or the MVP and is usually located on the next-to-last track of the third-from-last cylinder. An absolute sector pointer to its location is inserted in word 3 of track label 0.

Flaw definition information is also contained elsewhere on disk in a format different from the media deallocation data. For UDP/DP II, this alternate format is called *flaw data* and is located on the MD track. For HSDP, this is the *vendor defect table entries* and is located on the track next to the MD track.

Media flaw map is a text file that contains media flaw information in ASCII. This map can be written to SLO by specifying the DUMP or INCLUDE option with the INITIALIZE or EDITFMAP directives. Note that the HSDP uses this same term to refer to the contents of the MD track.

Dumped media flaw data is media flaw information written to tape or floppy disk in a format common to all media management processors (i.e., HSDP MVP, UDP/DP II MVP, and J.VFMT). The HSDP MVP refers to this format as the *dumped vendor defect table* and the UDP/DP II MVP refers to it as *saved flaw data.*

13.16.2 Initializing Media

Disks can be initialized in preparation for volume formatting by using the INITIALIZE directive. This directive writes new labels to a disk's tracks and sectors and builds the MD track.

If media flaw information is available for the disk, J.VFMT uses it when building the new MD track. This media flaw information may be either vendor flaw information input to the directive or an MD track already existing on the disk.

If media flaw information is unavailable, INITIALIZE writes a worst-case data pattern to the disk and reads the disk to check it for errors. If any additional media flaws are detected, J.VFMT adds them to the new MD track.

The INITIALIZE directive does not initialize the two cylinders reserved for diagnostic use or the cylinder reserved for hardware vendor use.

13.16.2.1 Disk I/O Caution During Initialization

With UDP/DP II, tasks should not attempt disk I/O to another channel subaddress while the controller is processing an INITIALIZE directive. Tasks attempting such I/O may be aborted or cause the system to hang. If running on another device subaddress on the system disk channel, hold the date/time save task (J.DTSAVE) with the OPCOM HOLD directive before initializing a disk.

If J.VFMT detects that I/O may be attempted to another channel subaddress, it issues the following warning message:

```
PRECAUTION:  ENSURE THAT ALL I/O TO OTHER DISC
SUBADDRESSES ON CHANNEL xx IS QUIESCENT BEFORE
CONTINUING
```

When executing in interactive mode, this is followed by a prompt to continue or abort processing. Enter `Y` to continue processing or `N` to abort with a VF01 error and message GEN.M062. When in batch mode, the following warning message is written to SLO and processing aborts:

```
BATCH MODE EXECUTION OF INITIALIZE DIRECTIVE TO
UDP/DPII MEDIA DENIED.
POTENTIAL FOR OTHER I/O ACTIVITY ON TARGET DISC CHANNEL
DETECTED.
```

13.16.3 Editing Media Flaw Data

Flaw definition information can be converted from one medium to another with the EDITFMAP directive. This directive executes in two modes: read-only mode and no I/O mode.

- Use read-only mode when the disk is present and has flaw information available for read-only access.
- Use no I/O mode when the disk cannot or need not be read.

13.16.3.1 Read-Only Mode

To specify read-only mode, include both the DISC and DEVICE options with the EDITFMAP directive.

With HSDP, this mode is used primarily to read vendor labels from a new disk without writing to the disk. In addition, this mode is useful when archiving, since vendor flaw information can be written to an external device in dumped media flaw format. It can also be used to obtain a hardcopy listing of a disk's vendor flaw information to compare to its vendor-supplied hardcopy listing.

With UDP and DP II, read-only mode is used primarily to perform a trial pass on the disk's existing MD track before using the INITIALIZE directive. Perform this trial pass before initializing any disk verified with the diagnostic MVP. Because previous MVP revisions for UDP/DP II vary in how they handle entries to the MD track, the INITIALIZE directive is not compatible with all MVP revisions. Use the EDITFMAP directive to identify discrepancies between the MVP-built MD track and J.VFMT's interpretation of this information.

If discrepancies are detected, do not use the information in the MVP-built MD track. Instead, input either vendor flaw information or the dumped media flaw data to the INITIALIZE directive. To do this, specify either the LOAD or INCLUDE option with the INITIALIZE directive.

13.16.3.2 No I/O Mode

To specify no I/O mode, include the DISC option and omit the DEVICE option with the EDITFMAP directive.

This mode is used primarily with the UDP/DP II or for removable media. Because these cannot read vendor flaw labels, you must manually enter vendor flaw information from a hardcopy vendor flaw listing supplied with the disk. Enter this vendor flaw information using a text editor and store it unnumbered in an MPX-32 file. Then use the EDITFMAP directive to process this input file and produce the requested outputs.

No I/O mode is useful for converting media flaw information from one format to another. For example, it can convert ASCII vendor flaw information in a file to dumped media flaw data on a floppy disk. The floppy disk could then be used to enter media flaw data during an SDT boot to initialize and format a single-disk UDP system.

13.16.4 Special Considerations During SDT Boot

The EDITFMAP directive is not available during SDT boot. Any attempt to use it causes processing to end and an error message is written to SLO.

The INITIALIZE directive is available but has the following considerations:

- the PATTERN option is not available
- the FMAP option writes SLO directly to an available line printer, instead of spooling it to disk
- LOAD and DUMP must specify different device addresses if both options are included with the directive
- devices specified in the LOAD and DUMP options must be online, ready, and correctly specified before you respond CONFIRM=N

13.16.5 Managing SCSI Disks

Media management of SCSI disks differs from media management of UDP/DP II and HSDP media as follows:

- the EDITFMAP directive is not available for SCSI disks. The intelligent SCSI disk controller performs this function differently and independently.
- CDC Wren III, IV, and V disks can be initialized using the J.VFMT INITIALIZE directive and a subset of its options. Options available with SCSI disks are DEVICE, DESTROY, and DISC. Following is an example using the SCSI syntax:

```
INI DEV=DM7E00 DES=Y DISC=SD0700
```

- SCSI disks other than CDC Wren III or IV can be initialized using the SCSI Disk Utility's (SDUTIL's) FORMAT directive. Refer to the SDUTIL documentation in Volume IV of the MPX-32 Reference Manual for this utility's syntax and use.

13.16.6 Media Flaw Map Format

Media flaw map format is an ASCII format that enables you to track the exact status of the disk media and to perform media maintenance. The media flaw map can be output to a file, edited, and then input to a media management function.

The media flaw map is comprised of flaw definition entries listed one per line in a format similar to the hardcopy listing provided by the vendor. Each flaw definition must be numerically higher than the previous one. In each definition:

- all values are decimal numbers
- an asterisk (*) indicates that the entry was flagged as a defective track by the vendor
- a C as the first nonblank character indicates a comment line. These can be included anywhere in a flaw map file.
- any number of blanks may precede the entry or separate the fields within it

The format for a flaw definition is:

[*] *cylnum* {bl, } *hdnum* {bl, } *posi* {bl, } *len*

[*]	indicates that the entry is a vendor-defined defective track
{bl, }	required field delimiter, either one or more blanks or a comma
cylnum	the cylinder number of the flaw definition
hdnum	the head number of the flaw definition
posi	the byte position of the flaw from the track index
len	the length in bytes of the flaw definition

14 Volume Compress (J.DSCMP)

14.1 General Description

Volume Compress (J.DSCMP) moves resource segments on a mounted volume to make the volume's free space contiguous. It is a privileged system administrator task.

J.DSCMP consolidates free disk space by moving resource segments to one end of a mounted volume and free space to the other end. When activated, J.DSCMP calculates the total free space available on disk and uses it to set the size or location of the following logical structures:

- vacate area — the area that J.DSCMP attempts to clear of resource segments. It is located at the high end (high physical block numbers) of the disk and is equal in size to the total free space available.
- pack area — the area where J.DSCMP moves and packs the resource segments. It is located at the low end (low physical block numbers) of the disk and is equal in size to the total allocated space on the disk.
- bound — separates the vacate area and the pack area. J.DSCMP calculates the location of the bound as the highest physical block number minus the total free space available on disk.

After locating the bound, J.DSCMP compresses a volume in two phases.

14.1.1 Phase One

In phase one, J.DSCMP attempts to fill the available contiguous free space in the pack area with resource segments from the vacate area. To do this, J.DSCMP compiles a V list (list of the resource segments in the vacate area), ordered from largest to smallest. For each segment in the V list, if there is sufficient contiguous free space in the pack area, J.DSCMP moves the segment below the bound and removes the segment from the V list. If the contiguous free space is insufficient, the segment remains in the V list for phase two. Phase one ends when J.DSCMP reaches the end of the V list.

If the V list is empty, compression is complete. Otherwise, phase two begins.

14.1.2 Phase Two

In phase two, J.DSCMP moves segments within the pack area to the low end of the disk until sufficient free space is available to move a segment in the V list below the bound. This consolidates resource segments at the low end of the disk and free space above the bound. Phase two continues until the V list is empty or the migrating free space moves above the segment with the highest physical address in the V list.

During compress, J.DSCMP does not move a resource segment if the segment:

- is allocated
- is not a permanent file (i.e., is a static or dynamic partition)
- was created with a specific start sector
- contains the RD for a directory
- contains the default image
- is being restored
- is a fast file
- has an entry in the memory resident descriptor table (MDT)
- is larger than the available free space in the pack area

The bound is adjusted for entries in the SMAP deallocation file. For every segment entry that is within the vacate area, the bound is lowered by that segment size.

J.DSCMP is a multi-copy task. However, only one copy at a time should be run on the same volume.

J.DSCMP uses standard MPX-32 I/O services to manipulate the data on the disk. All allocations and I/Os are performed in the wait mode with immediate denial.

14.1.3 Performance

Following is a list of performance considerations:

- J.DSCMP can be run while the volume is in use. However, compression is most effective when volume use is minimal.
- When compressing the system disk, compression results are better if M.ERR, M.PRJCT, M.ACCNT and M.KEY are not allocated. To do this, set control switch 4 when rebooting the system.
- Compression results are better if the disk to be compressed does not contain the swap file. Remove the swap file by specifying a swap file size of 0 at SYSGEN.

14.2 Logical File Code Assignments

Logical file codes required by J.DSCMP are summarized in Table 14-1.

Table 14-1
Volume Compress Logical File Code Assignments

Description	Logical File Code	Default Assignments for J.DSCMP
Error messages	ERR	$ASSIGN ERR TO LFC=UT
Informative messages	OUT	$ASSIGN OUT TO LFC=UT

14.3 Using Volume Compress

J.DSCMP is called from TSM.

Syntax

$J.DSCMP *volname* [**STAT** | [**LIST**][**NOMOVE**]]

volname	is the name of the mounted volume to be compressed
STAT	displays the current disk status. The volume is not compressed.
LIST	requests a list of moved segments (V and P segments) while compressing the volume. See Figure 14-1 for a sample list.
NOMOVE	requests a list of nonmovable segments (N segments) while compressing the volume. The segments remain in the vacate area.

14.3.1 Disk Status Report

J.DSCMP writes a disk status report to LFC OUT before and after compressing the volume or when the STAT option is specified in the J.DSCMP command. The disk status is displayed in the following format:

```
VOLUME= volume

TOTAL NUMBER OF BLOCKS (DEC)                          blocks
TOTAL FREE SPACE (DEC,PCT OF TOT)                     total   tpercent
LARGEST CONTIGUOUS FREE AREA (DEC,PCT OF FREE)        free    fpercent
```

blocks	is the decimal number of blocks on the disk
total	is the decimal number of free blocks on disk
tpercent	is the percent of blocks on disk that are free
free	is the decimal number of blocks in the largest contiguous free area
fpercent	is the percent of free blocks on disk that are in the largest contiguous free area. 100 indicates that all free space is contiguous.

14.3.2 Segment Report

If the LIST and/or NOMOVE options are specified, J.DSCMP displays a list of moved and/or nonmovable segments during volume compression. For each resource segment, the segment report lists the following:

- type (TYP) code that identifies whether and where the segment was moved
- original beginning and end sector numbers
- post-move beginning and end sector numbers
- RD address
- segment definition of the RD that was moved
- directory and resource where segment is located

Figure 14-1 shows a sample segment report that was requested specifying both LIST and NOMOVE. If only LIST is specified, the report lists only the V and P segments. If NOMOVE is specified, the report lists the N segments.

TYP	ORIG BEG	ORIG END	NEW BEG	NEW END	RD ADDR	SEG	DIRECTORY	RESOURCE
N	00013FEC	0001404F	MED. MAN. SEGMENT					
N	00014050	00014117	MED. MAN. SEGMENT					
N	00014118	0001417B	MED. MAN. SEGMENT					
N	000012E6	000012EF	NOT PERM. FILE		00000026	01	^	SOURCE
V	0000126E	000012A5	00000F0E	00000F45	00000024	01	TEST.02	CH.TAMM
P	00000E06	00000E4D	00000DD6	00000E1D	00000015	01	OBJECT_E	OH.PTRAC
P	00000E4E	00000EC5	00000E1E	00000E95	00000016	01	OBJECT_E	OH.REMM
P	00000EC6	00000F0D	00000E96	00000EDD	00000017	01	OBJECT_E	OH.REXS
P	00000F0E	00000F45	00000EDE	00000F15	00000024	01	TEST.02	OH.TAMM
P	00000F46	00000F85	00000F16	00000F55	00000019	01	OBJECT_E	OH.VOMM
V	000012A6	000012E5	00000F56	00000F95	00000025	01	TEST.02	OH.VOMM
P	00000FB6	00000FFD	00000F96	00000FDD	0000001B	01	TEST.01	OH.PTRAC
P	00000FFE	00001075	00000FDE	00001055	0000001C	01	TEST.01	OH.REMM
P	00001076	000010BD	00001056	0000109D	0000001D	01	TEST.01	OH.REXS
P	000010BE	000010F5	0000109E	000010D5	0000001E	01	TEST.01	OH.TAMM
P	000010F6	00001135	000010D6	00001115	0000001F	01	TEST.01	OH.VOMM
V	00001226	0000126D	00001116	0000115D	00000023	01	TEST.02	OH.REXS

Figure 14-1
J.DSCMP Segment Report

14.3.3 TYP Codes

After compression, J.DSCMP identifies each resource segment with one of the following TYP codes:

- N — Nonmovable segment that remains in the vacate area. J.DSCMP lists the reason why the segment was not moved in the NEW BEG column.

Message	Means that the segment
`A DEFAULT IMAGE`	contains a default image
`A FAST FILE`	is a fast file
`A MDT FILE`	has an entry in the MDT
`A RESTORING FILE`	is being restored
`FAIL CREATE TEMP`	temporary file couldn't be created
`FAIL TO OPEN FILE`	cannot be opened by J.DSCMP
`INVALID RD ADDRESS RD`	address doesn't match address in RID
`MED. MAN. SEGMENT`	used by Media Management (includes bad sectors)
`NOT PERM. FILE`	is not a permanent file
`START SECTOR FILE`	was created with a specific start sector
`WRONG SEGMENT-DEF`	definition changed since the V list was created
`WRONG TEMP SIZE`	temporary file is not large enough

- V — V list segment that was successfully moved from the vacate area to the pack area. The new segment definition (start and end sectors) is always below the bound.
- P — segment moved in phase two when consolidating segments towards the low end of the disk. If sufficient free space is not available in the pack area, P moves occur above the bound in the vacate area.

14.4 Error Messages

J.DSCMP displays the following error messages on the terminal:

```
FIRST PARAMETER MUST BE VOLUME NAME

SPECIFIED VOLUME IS NOT MOUNTED

SYSTEM ADMINISTRATOR ATTRIBUTE IS REQUIRED TO RUN THIS TASK

TOO MANY ARGUMENTS OR INVALID ARGUMENT SPECIFIED

UNABLE TO READ VOLUME DESCRIPTOR

UNABLE TO OBTAIN EXTENDED MEMORY

UNABLE TO READ SMAP DESCRIPTOR

UNABLE TO READ SMAP DEALLOCATION FILE DESCRIPTOR

UNABLE TO READ GENERAL ALLOCATABLE RESOURCE DESCRIPTORS'
DESCRIPTOR

UNABLE TO ASSIGN SMAP OR READ ERROR ON SMAP
```

The following messages display when an I/O error occurs while correcting a previous I/O error. Because the specified resource segment is no longer marked as allocated in the SMAP, it is susceptible to allocation to another resource if volume cleanup is not performed.

COMPRESSION TERMINATED DUE TO FATAL WRITE ERROR ON SEGMENT *x* (DECIMAL) OF PATH @*pathname*

FATAL ERROR. J.DSCMP SUSPENDED UNTIL REPAIR AND/OR REBOOT PERFORMED. @*pathname*

x is the segment where the error occurred

pathname is the volume, directory, and file where the error occurred

If this occurs, J.DSCMP remains in suspension and retains exclusive allocation of all relevant disk data until the system administrator performs a volume clean-up with control switch 7 not set. This corrects the file overlap by deleting the temporary file's resource descriptor without modifying the segment definition.

Note: If J.DSCMP exits or is deleted, the space used by the permanent file is freed.

A MPX-32 Device Access

A.1 Description

Throughout the MPX-32 Reference Manual, the generic descriptor *devmnc* indicates that a device can be specified.

Under MPX-32, device addresses are specified using a combination of three levels of identification. They are device type, device channel/controller address, and device address/subaddress.

A device can be specified using the generic device type mnemonic only, which results in allocation of the first available device of the type requested. Device type mnemonics are listed in Table A-1.

A second method of device specification is achieved by using the generic device type mnemonic and specifying the channel/controller address. This results in allocation of the first available device of the type requested on the specified channel or controller.

The third method of device selection requires specification of the device type mnemonic, channel/controller, and device address/subaddress. This method allows specification of a particular device.

Table A-1
Device Type Mnemonics and Codes

Device Type Code	Device Type Mnemonic	Device Description
00	CT	Operator console (not assignable)
01	DC	Any disk unit except memory disk
02	DM	Any moving head or memory disk
03	DF	Any fixed head disk
04	MT	Any magnetic tape unit
05	M9	Any 9-track magnetic tape unit*
06	M7	Any 7-track magnetic tape unit*
08	CR	Any card reader
0A	LP	Any line printer
0B	PT	Any paper tape reader-punch
0C	TY	Any teletypewriter (other than console)
0D	CT	Operator console (assignable)
0E	FL	Floppy disk
0F	NU	Null device
10	CA	Communications adapter (binary synchronous/asynchronous)
11	U0	Available for user-defined applications
12	U1	Available for user-defined applications
13	U2	Available for user-defined applications
14	U3	Available for user-defined applications
15	U4	Available for user-defined applications
16	U5	Available for user-defined applications
17	U6	Available for user-defined applications
18	U7	Available for user-defined applications
19	U8	Available for user-defined applications
1A	U9	Available for user-defined applications
1B	LF	Line printer/floppy controller (used only with SYSGEN)
N/A	ANY	Any nonfloppy disk except memory disk

* When both 7- and 9-track magnetic tape units are configured, the designation must be 7-track.

A.2 Special Device Specifications and Handling

A.2.1 Magnetic Tape/Floppy Disk

For magnetic tape and floppy disks, unblocking, density, a reel identifier, and multivolume number (magnetic tape only) can be included in the device specification.

Syntax

$ASSIGN *lfc* **TO DEV=***devmnc* [**BLO**CKED={**Y** | **N**}]
[**DEN**SITY={**N** | **P** | **G** | **800** | **1600** | **6250**}] [**ID=***id*] [**MUL**TIVOL=*number*]

lfc is a 1- to 3-character logical file code

DEV=*devmnc*
devmnc is the device specification of a configured peripheral device (see the Description section)

[**BLO**CKED={**Y** | **N**}]
if Y is specified, medium is blocked. If N is specified, medium is not blocked. If not specified the default is blocked.

[**DEN**SITY={**N** | **P** | **G** | **800** | **1600** | **6250**}]
specifies density of high speed XIO tape. If not specified, the default is 6250 bpi. Values are as follows:

Value	Description
N or 800	indicates 800 bpi nonreturn to zero inverted (NRZI).
P or 1600	indicates 1600 bpi phase encoded (PE).
G or 6250	indicates 6250 bpi group coded recording (GCR). This is the default.

[**ID=***id*] *id* specifies a 1- to 4-character identifier for the reel. If not specified, the default is SCRA (scratch).

[**MUL**TIVOL=*number*]
number is a volume number. If multivolume tape, *number* must be specified. If not specified, the default is not multivolume (0). This option is not valid for use with floppy disks.

When the task that has an assignment to tape is activated, a mount message indicates the name of the task and other information on the system console:

```
MOUNT reel VOL volume ON devmnc
TASK taskname,taskno REPLY R,H,A, OR DEVICE:
jobno
```

reel specifies a 1- to 4-character identifier for the reel. If not specified, the default is SCRA (Scratch).

volume identifies the volume number to mount if multivolume tape

devmnc is the device mnemonic for the tape unit selected in response to the assignment. If a specific channel and subaddress are supplied in the assignment, the specific tape drive is selected and named in the message; otherwise, a unit is selected by the system and its complete address is named in the message.

jobno identifies the job by job number if the task is part of a batch job

taskname is the name of the task to which the tape is assigned

taskno is the task number assigned to the task by the system

`R,H,A, OR DEVICE`
the device listed in the message can be allocated and the task resumed (`R`), a different device can be selected (`DEVICE`), the task can be aborted (`A`), or the task can be held with the specified device deallocated (`H`). If an R response is given and a high speed XIO tape drive is being used, its density can be changed when the software select feature is enabled on the tape unit front panel. If specified, it overrides any specification made at assignment. Example usage: `RN`, `R1600`, etc.

Note: Do not insert blanks or commas.

Response:

To indicate the drive specified in the mount message is ready and proceed with the task, mount the tape on the drive and type `R` (resume), optionally followed by a density specification if the drive is a high speed XIO tape unit. To abort the task, type `A` (abort). To hold the task and deallocate the specified device, type `H` (hold). The task can be resumed by the OPCOM CONTINUE directive; at which time, a tape drive is selected by the system and the mount message redisplayed.

To select a tape drive other than the drive specified in the message, enter the mnemonic of the drive to be used. Any of the three levels of device identification can be used. The mount message is reissued. Mount the tape and type `R` if satisfactory, or if not satisfactory, abort, override, or hold as described.

Examples of the three methods of device specification follow:

Type 1 - Generic Device Class

```
$ASSIGN OUT TO DEV=M9 MUL=1 ID=MVOL
```

In this example, the device assigned to logical file code (LFC) `OUT` is any `9`-track tape unit on any channel. The multivolume reel number is `1`. The reel identifier is `MVOL` and the tape is blocked.

Type 2 - Generic Device Class and Channel/Controller

```
$ASSIGN OUT TO DEV=M910 ID=MVOL BLO=N
```

In this example, the device assigned to logical file code (LFC) `OUT` is the first available `9`-track tape unit on channel `10`. The specification is invalid if a 9-track tape unit does not exist on the channel. The reel identifier is `MVOL`. This is not a multivolume tape and is unblocked.

Type 3 - Specific Device Request

```
$ASSIGN OUT TO DEV=M91001
```

In this example, the device assigned to logical file code (LFC) `OUT` is the `9`-track tape unit `01` on channel `10`. The specification is invalid if unit 01 on channel 10 is not a 9-track tape. The tape reel identifier is SCRA. The tape is blocked and is not multivolume.

A.2.2 Temporary Disk Space

For a temporary disk file the following can be specified: size, blocking, printing or punching, and access.

Syntax

$ASSIGN *lfc* **TO TEMP**[=(*volname*)] [**ACC**ESS=([**R**EAD] [**W**RITE] [**M**ODIFY] [**U**PDATE] [**A**PPEND])]
[**BLO**CKED={**Y** | **N**}] [**PRI**NT | **PUN**CH] [**SIZ**E=*blocks*]

lfc — is a 1- to 3-character logical file code

TEMP[=(*volname*)]
: *volname* is the 1- to 16-character volume name where temporary space is allocated. If not specified, the default is the current working volume or any public volume.

[**ACC**ESS=([**R**EAD] [**W**RITE] [**M**ODIFY] [**U**PDATE] [**A**PPEND])]
: specifies the types of access for the file. If not specified, the default is the access specified at file creation.

[**BLO**CKED={**Y** | **N**}]
: if Y is specified, the file is blocked. If N is specified, the file is unblocked. If not specified, the default is blocked.

[**PRI**NT | **PUN**CH]
: indicates the file is to be printed (PRINT) or punched (PUNCH) after deassignment

[**SIZ**E=*blocks*]
: *blocks* is the number of 192-word blocks required. If not specified, the default is 16 blocks.

Examples

In the following example, the device assigned to logical file code (LFC) `OUT` is the current working volume or any public volume and the file prints to the SLO device after deassignment.

```
AS OUT TO TEM PRI
```

The following example designates the system volume as the device for the temporary blocked file.

```
AS OUT TO TEMP=(SYSTEM) BLO=Y
```

A.3 GPMC Devices

GPMC/GPDC device specifications follow the general structure just described. The terminal at subaddress `04` on GPMC 01 whose channel address is `20` would be identified as follows:

```
$AS DEV TO DEV=TY2004
```

A.4 Null Device

A special device type, NU, is available for null device specifications. Files accessed using this device type generate an end-of-file (EOF) when a read is attempted and normal completion when a write is attempted.

A.5 System Console

Logical file codes are assigned to the system console by using the device type CT.

A.6 Special System Files

There are four special mnemonics provided for access to special system files: SLO, SBO, SGO and SYC. These are assigned with the $ASSIGN statement, as in:

```
$ASSIGN OUT TO SLO
```

For nonbatch tasks, SLO and SBO files are allocated dynamically by the system and used to disk buffer output to a device selected automatically. For batch tasks, use of SLO and SBO files is identical, except that automatic selection of a device can be overridden by assigning a specific file or device.

A.7 Samples

A description of device selection possibilities is constructed as follows:

Disk

DC	Any disk except memory disk
DM	Any moving head or memory disk
DM08	Any moving head disk on channel 08
DM0801	Moving head disk 01 on channel 08
DM0002	Memory disk 02 on channel 00
DF	Any fixed head disk
DF04	Any fixed head disk on channel 04
DF0401	Fixed head disk 01 on channel 04

Tape

MT	Any magnetic tape
M9	Any 9-track magnetic tape
M910	Any 9-track magnetic tape on channel 10
M91002	9-track magnetic tape 02 on channel 10

Card Equipment

CR	Any card reader
CR78	Any card reader on channel 78
CR7800	Card reader 00 on channel 78

Line Printer

LP	Any line printer
LP7A	Any line printer on channel 7A
LP7A00	Line printer 00 on channel 7A
LP7EA0	Serial printer A0 on ACM channel 7E

B System Services Cross-Reference

B.1 Macro Name Listing

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.ACTV	Activate Task	1,X'52'	H.REXS,15	6.2
M_ACTV	Activate Task	1,X'52'	H.REXS,15	7.2
M.ADRS	Memory Address Inquiry	1,X'44'	H.REXS,3	6.2
M_ADRS	Memory Address Inquiry	1,X'44'	H.REXS,3	7.2
M_ADVANCE	Advance Record Advance File	1,X'33' 1,X'34'	H.IOCS,7 H.IOCS,8	7.2 7.2
M.ALOC	Allocate File or Peripheral Device	1,X'40'	H.MONS,21	6.4
M.ANYW	Wait for Any No-wait Operation Complete, Message Interrupt, or Break Interrupt	1,X'7C'	H.REXS,37	6.2
M_ANYWAIT	Wait for Any No-wait Operation Complete, Message Interrupt, or Break Interrupt	1,X'7C'	H.REXS,37	7.2
M_ASSIGN	Assign and Allocate Resource	2,X'52'	H.REXS,21	7.2
M.ASSN	Assign and Allocate Resource	2,X'52'	H.REXS,21	6.2
M.ASYNCH	Set Asynchronous Task Interrupt	1,X'1C'	H.REXS,68	6.2
M_ASYNCH	Set Asynchronous Task Interrupt	1,X'1C'	H.REXS,68	7.2
M_AWAITACTION	End Action Wait	1,X'1D'	H.EXEC,40	7.2
M.BACK	Backspace Record Backspace File	1,X'35' 1,X'36'	H.IOCS,9 H.IOCS,19	6.2 6.2
M_BACKSPACE	Backspace Record Backspace File	1,X'35' 1,X'36'	H.IOCS,9 H.IOCS,19	7.2 7.2
M.BATCH	Batch Job Entry	2,X'55'	H.REXS,27	6.2
M_BATCH	Batch Job Entry	2,X'55'	H.REXS,27	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.BBTIM	Acquire Current Date/Time in Byte Binary Format	2,X'50'	H.REXS,74	6.2
M_BBTIM	Acquire Current Date/Time in Byte Binary Format	2,X'50'	H.REXS,74	7.2
M.BORT	Abort Specified Task	1,X'56'	H.REXS,19	6.2
	Abort Self	1,X'57'	H.REXS,20	6.2
	Abort With Extended Message	1,X'62'	H.REXS,28	6.2
M_BORT	Abort Specified Task	1,X'56'	H.REXS,19	7.2
	Abort Self	1,X'57'	H.REXS,20	7.2
	Abort With Extended Message	1,X'62'	H.REXS,28	7.2
M.BRK	Break/Task Interrupt Link/Unlink	1,X'6E'	H.REXS,46	6.2
M_BRK	Break/Task Interrupt Link/Unlink	1,X'6E'	H.REXS,46	7.2
M.BRKXIT	Exit From Task Interrupt Level	1,X'70'	H.REXS,48	6.2
M_BRKXIT	Exit From Task Interrupt Level	N/A	N/A	7.2
M.BTIM	Acquire Current Date/Time in Binary Format	2,X'50'	H.REXS,74	6.2
M_BTIM	Acquire Current Date/Time in Binary Format	2,X'50'	H.REXS,74	7.2
M.CDJS	Submit Job from Disc File	1,X'61'	H.MONS,27	6.4
M_CHANPROGFCB	Execute Channel Program File Control Block	N/A	N/A	7.2
M.CLOSER	Close Resource	2,X'43'	H.REMM,22	6.2
M_CLOSER	Close Resource	2,X'43'	H.REMM,22	7.2
M.CLSE	Close File	1,X'39'	H.IOCS,23	6.2
M_CLSE	Close File	1,X'39'	H.IOCS,23	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.CMD	Get Command Line	2,X'61'	H.REXS,88	6.2
M_CMD	Get Command Line	2,X'61'	H.REXS,88	7.2
M.CONABB	Convert ASCII Date/Time to Byte Binary Format	2,X'51'	H.REXS,75	6.2
M_CONABB	Convert ASCII Date/Time to Byte Binary Format	2,X'51'	H.REXS,75	7.2
M.CONADB	Convert ASCII Decimal to Binary	1,X'28'	H.TSM,7	6.2
M_CONADB	Convert ASCII Decimal to Binary	1,X'28'	H.TSM,7	7.2
M.CONAHB	Convert ASCII Hex to Binary	1,X'29'	H.TSM,8	6.2
M_CONAHB	Convert ASCII Hex to Binary	1,X'29'	H.TSM,8	7.2
M.CONASB	Convert ASCII Date/Time to Standard Binary	2,X'51'	H.REXS,75	6.2
M_CONASB	Convert ASCII Date/Time to Standard Binary	2,X'51'	H.REXS,75	7.2
M.CONBAD	Convert Binary to ASCII Decimal	1,X'2A'	H.TSM,9	6.2
M_CONBAD	Convert Binary to ASCII Decimal	1,X'2A'	H.TSM,9	7.2
M.CONBAF	Convert Binary Date/Time to ASCII Format	2,X'51'	H.REXS,75	6.2
M_CONBAF	Convert Binary Date/Time to ASCII Format	2,X'51'	H.REXS,75	7.2
M.CONBAH	Convert Binary to ASCII Hex	1,X'2B'	H.TSM,10	6.2
M_CONBAH	Convert Binary to ASCII Hex	1,X'2B'	H.TSM,10	7.2
M.CONBBA	Convert Byte Binary Date/Time to ASCII	2,X'51'	H.REXS,75	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M_CONBBA	Convert Byte Binary Date/Time to ASCII	2,X'51'	H.REXS,75	7.2
M.CONBBY	Convert Binary Date/Time to Byte Binary	2,X'51'	H.REXS,75	6.2
M_CONBBY	Convert Binary Date/Time to Byte Binary	2,X'51'	H.REXS,75	7.2
M.CONBYB	Convert Byte Binary Date/Time to Binary	2,X'51'	H.REXS,75	6.2
M_CONBYB	Convert Byte Binary Date/Time to Binary	2,X'51'	H.REXS,75	7.2
M.CONN	Connect Task to Interrupt	1,X'4B'	H.REXS,10	6.2
M_CONN	Connect Task to Interrupt	1,X'4B'	H.REXS,10	7.2
M_CONSTRUCTPATH	Reconstruct Pathname	2,X'2F'	H.VOMM,16	7.2
M_CONVERTTIME	Convert Time	2,X'51'	H.REXS,75	7.2
M.CPERM	Create Permanent File	2,X'20'	H.VOMM,1	6.2
M.CREATE	Create Permanent File	1,X'75'	H.FISE,12	6.4
M_CREATEFCB	Create File Control Block	N/A	N/A	7.2
M_CREATEP	Create Permanent File	2,X'20'	H.VOMM,1	7.2
M_CREATET	Create Temporary File	2,X'21'	H.VOMM,2	7.2
M.CTIM	Convert System Date/Time Format	2,X'51'	H.REXS,75	6.2
M_CTIM	Convert System Date/Time Format	2,X'51'	H.REXS,75	7.2
M.CWAT	System Console Wait	1,X'3D'	H.IOCS,26	6.2
M_CWAT	System Console Wait	1,X'3D'	H.IOCS,26	7.2
M.DALC	Deallocate File or Peripheral Device	1,X'41'	H.MONS,22	6.4
M.DASN	Deassign and Deallocate Resource	2,X'53'	H.REXS,22	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.DATE	Date and Time Inquiry	1,X'15'	H.REXS,70	6.2
M_DATE	Date and Time Inquiry	1,X'15'	H.REXS,70	7.2
M_DEASSIGN	Deassign and Deallocate Resource	2,X'53'	H.REXS,22	7.2
M.DEBUG	Load and Execute Interactive Debugger	1,X'63'	H.REXS,29	6.2
M_DEBUG	Load and Execute Interactive Debugger	1,X'63'	H.REXS,29	7.2
M.DEFT	Change Defaults	2,X'27'	H.VOMM,8	6.2
M_DEFT	Change Defaults	2,X'27'	H.VOMM,8	7.2
M.DELETE	Delete Permanent File or Non-SYSGEN Memory Partition	1,X'77'	H.FISE,14	6.4
M_DELETER	Delete Resource	2,X'24'	H.VOMM,5	7.2
M.DELR	Delete Resource	2,X'24'	H.VOMM,5	6.2
M.DELTSK	Delete Task	1,X'5A'	H.REXS,31	6.2
M_DELTSK	Delete Task	1,X'5A'	H.REXS,31	7.2
M.DEVID	Get Device Mnemonic or Type Code	1,X'14'	H.REXS,71	6.2
M_DEVID	Get Device Mnemonic or Type Code	1,X'14'	H.REXS,71	7.2
M.DFCB	Create File Control Block	N/A	N/A	5.9.1
M.DIR	Create Directory	2,X'23'	H.VOMM,4	6.2
M_DIR	Create Directory	2,X'23'	H.VOMM,4	7.2
M.DISCON	Disconnect Task from Interrupt	1,X'5D'	H.REXS,38	6.2
M_DISCON	Disconnect Task from Interrupt	1,X'5D'	H.REXS,38	7.2
M_DISMOUNT	Dismount Volume	2,X'4A'	H.REMM,19	7.2
M.DLTT	Delete Timer Entry	1,X'47'	H.REXS,6	6.2
M_DLTT	Delete Timer Entry	1,X'47'	H.REXS,6	7.2
M.DMOUNT	Dismount Volume	2,X'4A'	H.REMM,19	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.DSMI	Disable Message Task Interrupt	1,X'2E'	H.REXS,57	6.2
M_DSMI	Disable Message Task Interrupt	1,X'2E'	H.REXS,57	7.2
M.DSUB	Disable User Break Interrupt	1,X'12'	H.REXS,73	6.2
M_DSUB	Disable User Break Interrupt	1,X'12'	H.REXS,73	7.2
M.DUMP	Memory Dump Request	1,X'4F'	H.REXS,12	6.2
M_DUMP	Memory Dump Request	1,X'4F'	H.REXS,12	7.2
M.EAWAIT	End Action Wait	1,X'1D'	H.EXEC,40	6.2
M.ENMI	Enable Message Task Interrupt	1,X'2F'	H.REXS,58	6.2
M_ENMI	Enable Message Task Interrupt	1,X'2F'	H.REXS,58	7.2
M.ENUB	Enable User Break Interrupt	1,X'13'	H.REXS,72	6.2
M_ENUB	Enable User Break Interrupt	1,X'13'	H.REXS,72	7.2
M.ENVRMT	Get Task Environment	2,X'5E'	H.REXS,85	6.2
M_ENVRMT	Get Task Environment	2,X'5E'	H.REXS,85	7.2
M.EXCL	Free Shared Memory	1,X'79'	H.ALOC,14	6.4
M.EXCLUDE	Exclude Memory Partition	2,X'41'	H.REMM,14	6.2
M_EXCLUDE	Exclude Shared Image	2,X'41'	H.REMM,14	7.2
M.EXIT	Terminate Task Execution	1,X'55'	H.REXS,18	6.2
M_EXIT	Terminate Task Execution	1,X'55'	H.REXS,18	7.2
M.EXTD	Extend File	2,X'25'	H.VOMM,6	6.2
M_EXTENDFILE	Extend File	2,X'25'	H.VOMM,6	7.2
M_EXTSTS	Exit With Status	2,X'5F'	H.REXS,86	7.2
M.FADD	Permanent File Address Inquiry	1,X'43'	H.MONS,2	6.4

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.FD	Free Dynamic Extended Indexed Data Space	1,X'6A'	H.REMM,9	6.2
M.FE	Free Dynamic Task Execution Space	1,X'68'	H.REMM,11	6.2
M.FILE	Open File	1,X'30'	H.IOCS,1	6.4
M_FREEMEMBYTES	Free Memory in Byte Increments	2,X'4C'	H.REMM,29	7.2
M.FSLR	Release Synchronization File Lock	1,X'24'	H.FISE,25	6.4
M.FSLS	Set Synchronization File Lock	1,X'23'	H.FISE,24	6.4
M.FWRD	Advance Record Advance File	1,X'33' 1,X'34'	H.IOCS,7 H.IOCS,8	6.2 6.2
M.FXLR	Release Exclusive File Lock	1,X'22'	H.FISE,23	6.4
M.FXLS	Set Exclusive File Lock	1,X'21'	H.FISE,22	6.4
M.GADRL	Get Address Limits	1,X'65'	H.REXS,41	6.2
M.GADRL2	Get Address Limits	2,X'7B'	H.REXS,80	6.2
M.GD	Get Dynamic Extended Data Space	1,X'69'	H.REMM,8	6.2
M.GDD	Get Dynamic Extended Discontiguous Data Space	2,X'7C'	H.MEMM,9	6.2
M.GE	Get Dynamic Task Execution Space	1,X'67'	H.REMM,10	6.2
M_GETCTX	Get User Context	2,X'70'	H.EXEC,41	7.2
M.GETDEF	Get Terminal Function Definition	2,X'7A'	H.TSM,15	6.2
M_GETDEF	Get Terminal Function Definition	2,X'7A'	H.TSM,15	7.2
M_GETMEMBYTES	Get Memory in Byte Increments	2,X'4B'	H.REMM,28	7.2
M_GETTIME	Get Current Date and Time	2,X'50'	H.REXS,74	7.2
M.GMSGP	Get Message Parameters	1,X'7A'	H.REXS,35	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M_GMSGP	Get Message Parameters	1,X'7A'	H.REXS,35	7.2
M.GRUNP	Get Run Parameters	1,X'7B'	H.REXS,36	6.2
M_GRUNP	Get Run Parameters	1,X'7B'	H.REXS,36	7.2
M.GTIM	Acquire System Date/Time in Any Format	2,X'50'	H.REXS,74	6.2
M_GTIM	Acquire System Date/Time in Any Format	2,X'50'	H.REXS,74	7.2
M.GTSAD	Get TSA Start Address	2,X'7D'	H.REXS,91	6.2
M_GTSAD	Get TSA Start Address	2,X'7D'	H.REXS,91	7.2
M.HOLD	Program Hold Request	1,X'58'	H.REXS,25	6.2
M_HOLD	Program Hold Request	1,X'58'	H.REXS,25	7.2
M.ID	Get Task Number	1,X'64'	H.REXS,32	6.2
M_ID	Get Task Number	1,X'64'	H.REXS,32	7.2
M.INCL	Get Shared Memory	1,X'72'	H.ALOC,13	6.4
M.INCLUDE	Include Memory Partition	2,X'40'	H.REMM,12	6.2
M_INCLUDE	Include Shared Image	2,X'40'	H.REMM,12	7.2
M_INQUIRER	Resource Inquiry	2,X'48'	H.REMM,27	7.2
M.INQUIRY	Resource Inquiry	2,X'48'	H.REMM,27	6.2
M.INT	Activate Task Interrupt	1,X'6F'	H.REXS,47	6.2
M_INT	Activate Task Interrupt	1,X'6F'	H.REXS,47	7.2
M.IPUBS	Set IPU Bias	2,X'5B'	H.REXS,82	6.2
M_IPUBS	Set IPU Bias	2,X'5B'	H.REXS,82	7.2
M_LIMITS	Get Base Mode Task Address Limits	2,X'5D'	H.REXS,84	7.2
M.LOC	Read Descriptor	2,X'2C'	H.VOMM,13	6.2
M.LOCK	Set Exclusive Resource Lock	2,X'44'	H.REMM,23	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M_LOCK	Set Exclusive Resource Lock	2,X'44'	H.REMM,23	7.2
M.LOG	Permanent File Log	1,X'73'	H.MONS,33	6.4
M.LOGR	Log Resource or Directory	2,X'29'	H.VOMM,10	6.2
M_LOGR	Log Resource or Directory	2,X'29'	H.VOMM,10	7.2
M.MEM	Create Memory Partition	2,X'22'	H.VOMM,3	6.2
M_MEM	Create Memory Partition	2,X'22'	H.VOMM,3	7.2
M.MEMB	Get Memory in Byte Increments	2,X'4B'	H.REMM,28	6.2
M.MEMFRE	Free Memory in Byte Increments	2,X'4C'	H.REMM,29	6.2
M.MOD	Modify Descriptor	2,X'2A'	H.VOMM,11	6.2
M_MOD	Modify Descriptor	2,X'2A'	H.VOMM,11	7.2
M.MODU	Modify Descriptor User Area	2,X'31'	H.VOMM,26	6.2
M_MODU	Modify Descriptor User Area	2,X'31'	H.VOMM,26	7.2
M.MOUNT	Mount Volume	2,X'49'	H.REMM,17	6.2
M_MOUNT	Mount Volume	2,X'49'	H.REMM,17	7.2
M.MOVE	Move Data to User Address	2,X'62'	H.REXS,89	6.2
M_MOVE	Move Data to User Address	2,X'62'	H.REXS,89	7.2
M.MYID	Get Task Number	1,X'64'	H.REXS,32	6.2
M_MYID	Get Task Number	1,X'64'	H.REXS,32	7.2
M.NEWRRS	Reformat RRS Entry	2,X'54'	H.REXS,76	6.2
M.OLAY	Load Overlay Segment	1,X'50'	H.REXS,13	6.2
	Load and Execute Overlay	1,X'51'	H.REXS,14	6.2
M.OPENR	Open Resource	2,X'42'	H.REMM,21	6.2
M_OPENR	Open Resource	2,X'42'	H.REMM,21	7.2
M_OPTIONDWORD	Task Option Doubleword Inquiry	2,X'C0'	H.REXS,95	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M_OPTIONWORD	Task Option Word Inquiry	1,X'4C'	H.REXS,24	7.2
M.OSREAD	Physical Memory Read	2,X'7E'	H.REXS,93	6.2
M_OSREAD	Physical Memory Read	2,X'7E'	H.REXS,93	7.2
M.OSWRIT	Physical Memory Write	2,X'AF'	H.REXS,94	6.2
M_OSWRIT	Physical Memory Write	2,X'AF'	H.REXS,94	7.2
M.PDEV	Physical Device Inquiry	1,X'42'	H.MONS,1	6.4
M.PERM	Change Temporary File to Permanent	1,X'76'	H.FISE,13	6.4
M.PGOD	Task Option Doubleword Inquiry	2,X'C0'	H.REXS,95	6.2
M.PGOW	Task Option Word Inquiry	1,X'4C'	H.REXS,24	6.2
M.PNAM	Reconstruct Pathname	2,X'2F'	H.VOMM,16	6.2
M.PNAMB	Convert Pathname to Pathname Block	2,X'2E'	H.VOMM,15	6.2
M_PNAMB	Convert Pathname to Pathname Block	2,X'2E'	H.VOMM,15	7.2
M.PRIL	Change Priority Level	1,X'4A'	H.REXS,9	6.2
M_PRIL	Change Priority Level	1,X'4A'	H.REXS,9	7.2
M.PRIV	Reinstate Privilege Mode to Privilege Task	2,X'57'	H.REXS,78	6.2
M_PRIVMODE	Reinstate Privilege Mode to Privilege Task	2,X'57'	H.REXS,78	7.2
M.PTSK	Parameter Task Activation	1,X'5F'	H.REXS,40	6.2
M_PTSK	Parameter Task Activation	1,X'5F'	H.REXS,40	7.2
M_PUTCTX	Put User Context	2,X'71'	H.EXEC,42	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.QATIM	Acquire Current Date/Time in ASCII Format	2,X'50'	H.REXS,74	6.2
M_QATIM	Acquire Current Date/Time in ASCII Format	2,X'50'	H.REXS,74	7.2
M.RADDR	Get Real Physical Address	1,X'0E'	H.REXS,90	6.2
M_RADDR	Get Real Physical Address	1,X'0E'	H.REXS,90	7.2
M.RCVR	Receive Message Link Address	1,X'6B'	H.REXS,43	6.2
M_RCVR	Receive Message Link Address	1,X'6B'	H.REXS,43	7.2
M.READ	Read Record	1,X'31'	H.IOCS,3	6.2
M_READ	Read Record	1,X'31'	H.IOCS,3	7.2
M_READD	Read Descriptor	2,X'2C'	H.VOMM,13	7.2
M.RELP	Release Dual-ported Disc/Set Dual-channel ACM Mode	1,X'27'	H.IOCS,27	6.2
M_RELP	Release Dual-ported Disc/Set Dual-channel ACM Mode	1,X'27'	H.IOCS,27	7.2
M.RENAM	Rename File	2,X'2D'	H.VOMM,14	6.2
M_RENAME	Rename File	2,X'2D'	H.VOMM,14	7.2
M.REPLAC	Replace Permanent File	2,X'30'	H.VOMM,23	6.2
M_REPLACE	Replace Permanent File	2,X'30'	H.VOMM,23	7.2
M.RESP	Reserve Dual-ported Disc/Set Single-channel ACM Mode	1,X'26'	H.IOCS,24	6.2
M_RESP	Reserve Dual-ported Disc/Set Single-channel ACM Mode	1,X'26'	H.IOCS,24	7.2
M_REWIND	Rewind File	1,X'37'	H.IOCS,2	7.2
M.REWRIT	Rewrite Descriptor	2,X'2B'	H.VOMM,12	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M_REWRIT	Rewrite Descriptor	2,X'2B'	H.VOMM,12	7.2
M.REWRTU	Rewrite Descriptor User Area	2,X'32'	H.VOMM,27	6.2
M_REWRTU	Rewrite Descriptor User Area	2,X'32'	H.VOMM,27	7.2
M.ROPL	Reset Option Lower	2,X'78'	H.TSM,14	6.2
M_ROPL	Reset Option Lower	2,X'78'	H.TSM,14	7.2
M.RRES	Release Channel Reservation	1,X'3B'	H.IOCS,13	6.2
M_RRES	Release Channel Reservation	1,X'3B'	H.IOCS,13	7.2
M.RSML	Resourcemark Lock	1,X'19'	H.REXS,62	6.2
M_RSML	Resourcemark Lock	1,X'19'	H.REXS,62	7.2
M.RSMU	Resourcemark Unlock	1,X'1A'	H.REXS,63	6.2
M_RSMU	Resourcemark Unlock	1,X'1A'	H.REXS,63	7.2
M.RSRV	Reserve Channel	1,X'3A'	H.IOCS,12	6.2
M_RSRV	Reserve Channel	1,X'3A'	H.IOCS,12	7.2
M.RWND	Rewind File	1,X'37'	H.IOCS,2	6.2
M_SETERA	Set Exception Return Address	2,X'79'	H.REXS,81	7.2
M_SETEXA	Set Exception Handler	2,X'5C'	H.REXS,83	7.2
M.SETS	Set User Status Word	1,X'48'	H.REXS,7	6.2
M_SETS	Set User Status Word	1,X'48'	H.REXS,7	7.2
M.SETSYNC	Set Synchronous Resource Lock	2,X'46'	H.REMM,25	6.2
M_SETSYNC	Set Synchronous Resource Lock	2,X'46'	H.REMM,25	7.2
M.SETT	Create Timer Entry	1,X'45'	H.REXS,4	6.2
M_SETT	Create Timer Entry	1,X'45'	H.REXS,4	7.2
M.SHARE	Share Memory with Another Task	1,X'71'	H.ALOC,12	6.4
M.SMSGR	Send Message to Specified Task	1,X'6C'	H.REXS,44	6.2
M_SMSGR	Send Message to Specified Task	1,X'6C'	H.REXS,44	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.SMULK	Unlock and Dequeue Shared Memory	1,X'1F'	H.ALOC,19	6.4
M.SOPL	Set Option Lower	2,X'77'	H.TSM,13	6.2
M_SOPL	Set Option Lower	2,X'77'	H.TSM,13	7.2
M.SRUNR	Send Run Request to Specified Task	1,X'6D'	H.REXS,45	6.2
M_SRUNR	Send Run Request to Specified Task	1,X'6D'	H.REXS,45	7.2
M.SUAR	Set User Abort Receiver Address	1,X'60'	H.REXS,26	6.2
M_SUAR	Set User Abort Receiver Address	1,X'60'	H.REXS,26	7.2
M.SUME	Resume Task Execution	1,X'53'	H.REXS,16	6.2
M_SUME	Resume Task Execution	1,X'53'	H.REXS,16	7.2
M.SURE	Suspend/Resume	5,X'00'	N/A	6.2
M_SURE	Suspend/Resume	5,X'00'	N/A	7.2
M.SUSP	Suspend Task Execution	1,X'54'	H.REXS,17	6.2
M_SUSP	Suspend Task Execution	1,X'54'	H.REXS,17	7.2
M.SYNCH	Set Synchronous Task Interrupt	1,X'1B'	H.REXS,67	6.2
M_SYNCH	Set Synchronous Task Interrupt	1,X'1B'	H.REXS,67	7.2
M.TBRKON	Trap On-line User's Task	1,X'5C'	H.TSM,6	6.2
M_TBRKON	Trap On-line User's Task	1,X'5C'	H.TSM,6	7.2
M.TDAY	Time-of-Day Inquiry	1,X'4E'	H.REXS,11	6.2
M_TDAY	Time-of-Day Inquiry	1,X'4E'	H.REXS,11	7.2
M.TEMP	Create Temporary File	2,X'21'	H.VOMM,2	6.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.TEMPER	Change Temporary File to Permanent File	2,X'28'	H.VOMM,9	6.2
M_TEMPFILETOPERM	Change Temporary File to Permanent File	2,X'28'	H.VOMM,9	7.2
M.TRNC	Truncate File	2,X'26'	H.VOMM,7	6.2
M_TRUNCATE	Truncate File	2,X'26'	H.VOMM,7	7.2
M.TSCAN	Scan Terminal Input Buffer	1,X'5B'	H.TSM,2	6.2
M_TSCAN	Scan Terminal Input Buffer	1,X'5B'	H.TSM,2	7.2
M.TSMPC	TSM Procedure Call	2,X'AE'	H.TSM,17	6.2
M_TSMPC	TSM Procedure Call	2,X'AE'	H.TSM,17	7.2
M.TSTE	Arithmetic Exception Inquiry	1,X'4D'	H.REXS,23	6.2
M_TSTE	Arithmetic Exception Inquiry	1,X'4D'	H.REXS,23	7.2
M.TSTS	Test User Status Word	1,X'49'	H.REXS,8	6.2
M_TSTS	Test User Status Word	1,X'49'	H.REXS,8	7.2
M.TSTT	Test Timer Entry	1,X'46'	H.REXS,5	6.2
M_TSTT	Test Timer Entry	1,X'46'	H.REXS,5	7.2
M.TURNON	Activate Program at Given Time of Day	1,X'1E'	H.REXS,66	6.2
M_TURNON	Activate Program at Given Time of Day	1,X'1E'	H.REXS,66	7.2
M.TYPE	System Console Type	1,X'3F'	H.IOCS,14	6.2
M_TYPE	System Console Type	1,X'3F'	H.IOCS,14	7.2
M.UNLOCK	Release Exclusive Resource Lock	2,X'45'	H.REMM,24	6.2
M_UNLOCK	Release Exclusive Resource Lock	2,X'45'	H.REMM,24	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M_UNPRIVMODE	Change Task to Unprivileged Mode	2,X'58'	H.REXS,79	7.2
M.UNSYNC	Release Synchronous Resource Lock	2,X'47'	H.REMM,26	6.2
M_UNSYNC	Release Synchronous Resource Lock	2,X'47'	H.REMM,26	7.2
M.UPRIV	Change Task to Unprivileged Mode	2,X'58'	H.REXS,79	6.2
M.UPSP	Upspace	1,X'10'	H.IOCS,20	6.2
M_UPSP	Upspace	1,X'10'	H.IOCS,20	7.2
M.USER	User Name Specification	1,X'74'	H.MONS,34	6.4
M.VADDR	Validate Address Range	2,X'59'	H.REXS,33	6.2
M_VADDR	Validate Address Range	2,X'59'	H.REXS,33	7.2
M.WAIT	Wait I/O	1,X'3C'	H.IOCS,25	6.2
M_WAIT	Wait I/O	1,X'3C'	H.IOCS,25	7.2
M.WEOF	Write EOF	1,X'38'	H.IOCS,5	6.2
M.WRIT	Write Record	1,X'32'	H.IOCS,4	6.2
M_WRITE	Write Record	1,X'32'	H.IOCS,4	7.2
M_WRITEEOF	Write EOF	1,X'38'	H.IOCS,5	7.2
M.XBRKR	Exit from Task Interrupt Level	1,X'70'	H.REXS,48	6.2
M_XBRKR	Exit from Task Interrupt Level	N/A	N/A	7.2
M.XIEA	No-wait I/O End-action Return	1,X'2C'	H.IOCS,34	6.2
M_XIEA	No-wait I/O End-action Return	N/A	N/A	7.2
M.XMEA	Exit from Message End-action Routine	1,X'7E'	H.REXS,50	6.2
M_XMEA	Exit from Message End-action Routine	N/A	N/A	7.2

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
M.XMSGR	Exit from Message Receiver	1,X'5E'	H.REXS,39	6.2
M_XMSGR	Exit from Message Receiver	N/A	N/A	7.2
M.XREA	Exit from Run Request End-action Routine	1,X'7F'	H.REXS,51	6.2
M_XREA	Exit from Run Request End-action Routine	N/A	N/A	7.2
M.XRUNR	Exit Run Receiver	1,X'7D'	H.REXS,49	6.2
M_XRUNR	Exit Run Receiver	N/A	N/A	7.2
M.XTIME	Task CPU Execution Time	1,X'2D'	H.REXS,65	6.2
M_XTIME	Task CPU Execution Time	1,X'2D'	H.REXS,65	7.2
N/A	Allocate File Space	N/A	H.VOMM,19	6.3
N/A	Allocate Resource Descriptor	N/A	H.VOMM,17	6.3
N/A	Create Temporary File	N/A	H.VOMM,24	6.3
N/A	Deallocate File Space	N/A	H.VOMM,20	6.3
N/A	Deallocate Resource Descriptor	N/A	H.VOMM,18	6.3
N/A	Debug Link Service	1,X'66'	H.REXS,42	6.3
N/A	Debug Link Service-Base Mode	1,X'66'	H.REXS,42	7.3
N/A	Eject/Purge Routine	1,X'0D'	H.IOCS,22	6.3
N/A	Eject/Purge Routine-Base Mode	1,X'0D'	H.IOCS,22	7.3
N/A	Erase or Punch Trailer	1,X'3E'	H.IOCS,21	6.3
N/A	Erase or Punch Trailer - Base Mode	1,X'3E'	H.IOCS,21	7.3
N/A	Execute Channel Program	1,X'25'	H.IOCS,10	6.3
N/A	Execute Channel Program - Base Mode	1,X'25'	H.IOCS,10	7.3

Macro	Description	SVC	Module, E.P.	Volume I Ref.Manual Section
N/A	Get Extended Memory Array	2,X'7F'	H.MEMM,14	6.3
N/A	Get Extended Memory Array - Base Mode	2,X'7F'	H.MEMM,14	7.3
N/A	Read/Write Authorization File	N/A	H.VOMM,25	6.3
N/A	Release FHD Port	1,X'27'	H.IOCS,27	6.3
N/A	Release FHD Port - Base Mode	1,X'27'	H.IOCS,27	7.3
N/A	Reserve FHD Port	1,X'26'	H.IOCS,24	6.3
N/A	Reserve FHD Port- Base Mode	1,X'26'	H.IOCS,24	7.3
N/A	Reserved for Interactive Debugger	2,X'56'	H.REXS,30	N/A
N/A	Reserved for Rapid File Allocation:			N/A
	Zero MDT	2,X'AA'	H.MDT,1	
	Locate/Read MDT Entry	2,X'AB'	H.MDT,2	
	Update/Create MDT Entry	2,X'AC'	H.MDT,3	
	Delete MDT Entry	2,X'AD'	H.MDT,4	
N/A	Set Tabs in UDT	1,X'59'	H.TSM,5	N/A
N/A	TSM Task Detach	1,X'20'	H.TSM,3	N/A

B.2 Alphabetic Listing

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Abort Self	M.BORT M_BORT	1,X'57' 1,X'57'	H.REXS,20 H.REXS,20	6.2 7.2
Abort Specified Task	M.BORT M_BORT	1,X'56' 1,X'56'	H.REXS,19 H.REXS,19	6.2 7.2
Abort With Extended Message	M.BORT M_BORT	1,X'62' 1,X'62'	H.REXS,28 H.REXS,28	6.2 7.2
Acquire Current Date/Time in ASCII Format	M.QATIM M_QATIM	2,X'50' 2,X'50'	H.REXS,74 H.REXS,74	6.2 7.2
Acquire Current Date/Time in Binary Format	M.BTIM M_BTIM	2,X'50' 2,X'50'	H.REXS,74 H.REXS,74	6.2 7.2
Acquire Current Date/Time in Byte Binary Format	M.BBTIM M_BBTIM	2,X'50' 2,X'50'	H.REXS,74 H.REXS,74	6.2 7.2
Acquire System Date/Time in Any Format	M.GTIM M_GTIM	2,X'50' 2,X'50'	H.REXS,74 H.REXS,74	6.2 7.2
Activate Program at Given Time of Day	M.TURNON M_TURNON	1,X'1E' 1,X'1E'	H.REXS,66 H.REXS,66	6.2 7.2
Activate Task	M.ACTV M_ACTV	1,X'52' 1,X'52'	H.REXS,15 H.REXS,15	6.2 7.2
Activate Task Interrupt	M.INT M_INT	1,X'6F' 1,X'6F'	H.REXS,47 H.REXS,47	6.2 7.2
Advance File	M.FWRD M_ADVANCE	1,X'34' 1,X'34'	H.IOCS,8 H.IOCS,8	6.2 7.2
Advance Record	M.FWRD M_ADVANCE	1,X'33' 1,X'33'	H.IOCS,7 H.IOCS,7	6.2 7.2
Allocate File or Peripheral Device	M.ALOC	1,X'40'	H.MONS,21	6.4
Allocate File Space	N/A	N/A	H.VOMM,19	6.3
Allocate Resource Descriptor	N/A	N/A	H.VOMM,17	6.3
Arithmetic Exception Inquiry	M.TSTE M_TSTE	1,X'4D' 1,X'4D'	H.REXS,23 H.REXS,23	6.2 7.2

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Assign and Allocate Resource	M.ASSN	2,X'52'	H.REXS,21	6.2
	M_ASSIGN	2,X'52'	H.REXS,21	7.2
Backspace File	M.BACK	1,X'36'	H.IOCS,19	6.2
	M_BACKSPACE	1,X'36'	H.IOCS,19	7.2
Backspace Record	M.BACK	1,X'35'	H.IOCS,9	6.2
	M_BACKSPACE	1,X'35'	H.IOCS,9	7.2
Batch Job Entry	M.BATCH	2,X'55'	H.REXS,27	6.2
	M_BATCH	2,X'55'	H.REXS,27	7.2
Break/Task Interrupt Link/Unlink	M.BRK	1,X'6E'	H.REXS,46	6.2
	M_BRK	1,X'6E'	H.REXS,46	7.2
Change Defaults	M.DEFT	2,X'27'	H.VOMM,8	6.2
	M_DEFT	2,X'27'	H.VOMM,8	7.2
Change Priority Level	M.PRIL	1,X'4A'	H.REXS,9	6.2
	M_PRIL	1,X'4A'	H.REXS,9	7.2
Change Task to Unprivileged Mode	M.UPRIV	2,X'58'	H.REXS,79	6.2
	M_UNPRIVMODE	2,X'58'	H.REXS,79	7.2
Change Temporary File to Permanent	M.PERM	1,X'76'	H.FISE,13	6.4
Change Temporary File to Permanent File	M.TEMPER	2,X'28'	H.VOMM,9	6.2
	M_TEMPFILETOPERM	2,X'28'	H.VOMM,9	7.2
Close File	M.CLSE	1,X'39'	H.IOCS,23	6.2
	M_CLSE	1,X'39'	H.IOCS,23	7.2
Close Resource	M.CLOSER	2,X'43'	H.REMM,22	6.2
	M_CLOSER	2,X'43'	H.REMM,22	7.2
Connect Task to Interrupt	M.CONN	1,X'4B'	H.REXS,10	6.2
	M_CONN	1,X'4B'	H.REXS,10	7.2
Convert ASCII Date/Time to Byte Binary Format	M.CONABB	2,X'51'	H.REXS,75	6.2
	M_CONABB	2,X'51'	H.REXS,75	7.2
Convert ASCII Date/Time to Standard Binary	M.CONASB	2,X'51'	H.REXS,75	6.2
	M_CONASB	2,X'51'	H.REXS,75	7.2
Convert ASCII Decimal to Binary	M.CONADB	1,X'28'	H.TSM,7	6.2
	M_CONADB	1,X'28'	H.TSM,7	7.2
Convert ASCII Hex to Binary	M.CONAHB	1,X'29'	H.TSM,8	6.2
	M_CONAHB	1,X'29'	H.TSM,8	7.2

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Convert Binary Date/Time to ASCII Format	M.CONBAF M_CONBAF	2,X'51' 2,X'51'	H.REXS,75 H.REXS,75	6.2 7.2
Convert Binary Date/Time to Byte Binary	M.CONBBY M_CONBBY	2,X'51' 2,X'51'	H.REXS,75 H.REXS,75	6.2 7.2
Convert Binary to ASCII Decimal	M.CONBAD M_CONBAD	1,X'2A' 1,X'2A'	H.TSM,9 H.TSM,9	6.2 7.2
Convert Binary to ASCII Hex	M.CONBAH M_CONBAH	1,X'2B' 1,X'2B'	H.TSM,10 H.TSM,10	6.2 7.2
Convert Byte Binary Date/Time to ASCII	M.CONBBA M_CONBBA	2,X'51' 2,X'51'	H.REXS,75 H.REXS,75	6.2 7.2
Convert Byte Binary Date/Time to Binary	M.CONBYB M_CONBYB	2,X'51' 2,X'51'	H.REXS,75 H.REXS,75	6.2 7.2
Convert Pathname to Pathname Block	M.PNAMB M_PNAMB	2,X'2E' 2,X'2E'	H.VOMM,15 H.VOMM,15	6.2 7.2
Convert System Date/Time Format	M.CTIM M_CTIM	2,X'51' 2,X'51'	H.REXS,75 H.REXS,75	6.2 7.2
Convert Time	M_CONVERTTIME	2,X'51'	H.REXS,75	7.2
Create Directory	M.DIR M_DIR	2,X'23' 2,X'23'	H.VOMM,4 H.VOMM,4	6.2 7.2
Create File Control Block	M.DFCB	N/A	N/A	5.9.1
Create File Control Block	M_CREATEFCB	N/A	N/A	7.2
Create Memory Partition	M.MEM M_MEM	2,X'22' 2,X'22'	H.VOMM,3 H.VOMM,3	6.2 7.2
Create Permanent File	M.CREATE	1,X'75'	H.FISE,12	6.4
Create Permanent File	M.CPERM M_CREATEP	2,X'20' 2,X'20'	H.VOMM,1 H.VOMM,1	6.2 7.2
Create Temporary File	M.TEMP M_CREATET	2,X'21' 2,X'21'	H.VOMM,2 H.VOMM,2	6.2 7.2
Create Temporary File	N/A	N/A	H.VOMM,24	6.3

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Create Timer Entry	M.SETT M_SETT	1,X'45' 1,X'45'	H.REXS,4 H.REXS,4	6.2 7.2
Date and Time Inquiry	M.DATE M_DATE	1,X'15' 1,X'15'	H.REXS,70 H.REXS,70	6.2 7.2
Deallocate File or Peripheral Device	M.DALC	1,X'41'	H.MONS,22	6.4
Deallocate File Space	N/A	N/A	H.VOMM,20	6.3
Deallocate Resource Descriptor	N/A	N/A	H.VOMM,18	6.3
Deassign and Deallocate Resource	M.DASN M_DEASSIGN	2,X'53' 2,X'53'	H.REXS,22 H.REXS,22	6.2 7.2
Debug Link Service	N/A	1,X'66'	H.REXS,42	6.3
Debug Link Service-Base Mode	N/A	1,X'66'	H.REXS,42	7.3
Delete Permanent File or Non-SYSGEN Memory Partition	M.DELETE	1,X'77'	H.FISE,14	6.4
Delete Resource	M.DELR M_DELETER	2,X'24' 2,X'24'	H.VOMM,5 H.VOMM,5	6.2 7.2
Delete Task	M.DELTSK M_DELTSK	1,X'5A' 1,X'5A'	H.REXS,31 H.REXS,31	6.2 7.2
Delete Timer Entry	M.DLTT M_DLTT	1,X'47' 1,X'47'	H.REXS,6 H.REXS,6	6.2 7.2
Disable Message Task Interrupt	M.DSMI M_DSMI	1,X'2E' 1,X'2E'	H.REXS,57 H.REXS,57	6.2 7.2
Disable User Break Interrupt	M.DSUB M_DSUB	1,X'12' 1,X'12'	H.REXS,73 H.REXS,73	6.2 7.2
Disconnect Task from Interrupt	M.DISCON M_DISCON	1,X'5D' 1,X'5D'	H.REXS,38 H.REXS,38	6.2 7.2
Dismount Volume	M.DMOUNT M_DISMOUNT	2,X'4A' 2,X'4A'	H.REMM,19 H.REMM,19	6.2 7.2
Eject/Purge Routine	N/A	1,X'0D'	H.IOCS,22	6.3
Eject/Purge Routine-Base Mode	N/A	1,X'0D'	H.IOCS,22	7.3
Enable Message Task Interrupt	M.ENMI M_ENMI	1,X'2F' 1,X'2F'	H.REXS,58 H.REXS,58	6.2 7.2

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Enable User Break Interrupt	M.ENUB	1,X'13'	H.REXS,72	6.2
	M_ENUB	1,X'13'	H.REXS,72	7.2
End Action Wait	M.EAWAIT	1,X'1D'	H.EXEC,40	6.2
	M_AWAITACTION	1,X'1D'	H.EXEC,40	7.2
Erase or Punch Trailer	N/A	1,X'3E'	H.IOCS,21	6.3
Erase or Punch Trailer - Base Mode	N/A	1,X'3E'	H.IOCS,21	7.3
Exclude Memory Partition	M.EXCLUDE	2,X'41'	H.REMM,14	6.2
Exclude Shared Image	M_EXCLUDE	2,X'41'	H.REMM,14	7.2
Execute Channel Program	N/A	1,X'25'	H.IOCS,10	6.3
Execute Channel Program - Base Mode	N/A	1,X'25'	H.IOCS,10	7.3
Execute Channel Program File Control Block	M_CHANPROGFCB	N/A	N/A	7.2
Exit from Message End-action Routine	M.XMEA	1,X'7E'	H.REXS,50	6.2
	M_XMEA	N/A	N/A	7.2
Exit from Message Receiver	M.XMSGR	1,X'5E'	H.REXS,39	6.2
	M_XMSGR	N/A	N/A	7.2
Exit from Run Request End-action Routine	M.XREA	1,X'7F'	H.REXS,51	6.2
	M_XREA	N/A	N/A	7.2
Exit from Task Interrupt Level	M.BRKXIT	1,X'70'	H.REXS,48	6.2
	M _BRKXIT	N/A	N/A	7.2
	M.XBRKR	1,X'70'	H.REXS,48	6.2
	M _XBRKR	N/A	N/A	7.2
Exit Run Receiver	M.XRUNR	1,X'7D'	H.REXS,49	6.2
	M_XRUNR	N/A	N/A	7.2
Exit With Status	M_EXTSTS	2,X'5F'	H.REXS,86	7.2
Extend File	M.EXTD	2,X'25'	H.VOMM,6	6.2
	M_EXTENDFILE	2,X'25'	H.VOMM,6	7.2
Free Dynamic Extended Indexed Data Space	M.FD	1,X'6A'	H.REMM,9	6.2

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Free Dynamic Task Execution Space	M.FE	1,X'68'	H.REMM,11	6.2
Free Memory in Byte Increments	M.MEMFRE	2,X'4C'	H.REMM,29	6.2
	M_FREEMEMBYTES	2,X'4C'	H.REMM,29	7.2
Free Shared Memory	M.EXCL	1,X'79'	H.ALOC,14	6.4
Get Address Limits	M.GADRL	1,X'65'	H.REXS,41	6.2
Get Address Limits	M.GADRL2	2,X'7B'	H.REXS,80	6.2
Get Base Mode Task Address Limits	M_LIMITS	2,X'5D'	H.REXS,84	7.2
Get Command Line	M.CMD	2,X'61'	H.REXS,88	6.2
	M_CMD	2,X'61'	H.REXS,88	7.2
Get Current Date and Time	M_GETTIME	2,X'50'	H.REXS,74	7.2
Get Device Mnemonic or Type Code	M.DEVID	1,X'14'	H.REXS,71	6.2
	M_DEVID	1,X'14'	H.REXS,71	7.2
Get Dynamic Extended Data Space	M.GD	1,X'69'	H.REMM,8	6.2
Get Dynamic Extended Discontiguous Data Space	M.GDD	2,X'7C'	H.MEMM,9	6.2
Get Dynamic Task Execution Space	M.GE	1,X'67'	H.REMM,10	6.2
Get Extended Memory Array	N/A	2,X'7F'	H.MEMM,14	6.3
Get Extended Memory Array - Base Mode	N/A	2,X'7F'	H.MEMM,14	7.3
Get Memory in Byte Increments	M.MEMB	2,X'4B'	H.REMM,28	6.2
	M_GETMEMBYTES	2,X'4B'	H.REMM,28	7.2
Get Message Parameters	M.GMSGP	1,X'7A'	H.REXS,35	6.2
	M_GMSGP	1,X'7A'	H.REXS,35	7.2
Get Real Physical Address	M.RADDR	1,X'0E'	H.REXS,90	6.2
	M_RADDR	1,X'0E'	H.REXS,90	7.2
Get Run Parameters	M.GRUNP	1,X'7B'	H.REXS,36	6.2
	M_GRUNP	1,X'7B'	H.REXS,36	7.2
Get Shared Memory	M.INCL	1,X'72'	H.ALOC,13	6.4

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Get Task Environment	M.ENVRMT	2,X'5E'	H.REXS,85	6.2
	M_ENVRMT	2,X'5E'	H.REXS,85	7.2
Get Task Number	M.ID	1,X'64'	H.REXS,32	6.2
	M_ID	1,X'64'	H.REXS,32	7.2
	M.MYID	1,X'64'	H.REXS,32	6.2
	M_MYID	1,X'64'	H.REXS,32	7.2
Get Terminal Function Definition	M.GETDEF	2,X'7A'	H.TSM,15	6.2
	M_GETDEF	2,X'7A'	H.TSM,15	7.2
Get TSA Start Address	M.GTSAD	2,X'7D'	H.REXS,91	6.2
	M_GTSAD	2,X'7D'	H.REXS,91	7.2
Get User Context	M_GETCTX	2,X'70'	H.EXEC,41	7.2
Include Memory Partition	M.INCLUDE	2,X'40'	H.REMM,12	6.2
Include Shared Image	M_INCLUDE	2,X'40'	H.REMM,12	7.2
Load and Execute Interactive Debugger	M.DEBUG	1,X'63'	H.REXS,29	6.2
	M_DEBUG	1,X'63'	H.REXS,29	7.2
Load Overlay Segment	M.OLAY	1,X'50'	H.REXS,13	6.2
Load and Execute Overlay		1,X'51'	H.REXS,14	6.2
Log Resource or Directory	M.LOGR	2,X'29'	H.VOMM,10	6.2
	M_LOGR	2,X'29'	H.VOMM,10	7.2
Memory Address Inquiry	M.ADRS	1,X'44'	H.REXS,3	6.2
	M_ADRS	1,X'44'	H.REXS,3	7.2
Memory Dump Request	M.DUMP	1,X'4F'	H.REXS,12	6.2
	M_DUMP	1,X'4F'	H.REXS,12	7.2
Modify Descriptor	M.MOD	2,X'2A'	H.VOMM,11	6.2
	M_MOD	2,X'2A'	H.VOMM,11	7.2
Modify Descriptor User Area	M.MODU	2,X'31'	H.VOMM,26	6.2
	M_MODU	2,X'31'	H.VOMM,26	7.2
Mount Volume	M.MOUNT	2,X'49'	H.REMM,17	6.2
	M_MOUNT	2,X'49'	H.REMM,17	7.2
Move Data to User Address	M.MOVE	2,X'62'	H.REXS,89	6.2
	M_MOVE	2,X'62'	H.REXS,89	7.2
No-wait I/O End-action Return	M.XIEA	1,X'2C'	H.IOCS,34	6.2
	M_XIEA	N/A	N/A	7.2
Open File	M.FILE	1,X'30'	H.IOCS,1	6.4

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Open Resource	M.OPENR	2,X'42'	H.REMM,21	6.2
	M_OPENR	2,X'42'	H.REMM,21	7.2
Parameter Task Activation	M.PTSK	1,X'5F'	H.REXS,40	6.2
	M_PTSK	1,X'5F'	H.REXS,40	7.2
Permanent File Address Inquiry	M.FADD	1,X'43'	H.MONS,2	6.4
Permanent File Log	M.LOG	1,X'73'	H.MONS,33	6.4
Physical Device Inquiry	M.PDEV	1,X'42'	H.MONS,1	6.4
Physical Memory Read	M.OSREAD	2,X'7E'	H.REXS,93	6.2
	M_OSREAD	2,X'7E'	H.REXS,93	7.2
Physical Memory Write	M.OSWRIT	2,X'AF'	H.REXS,94	6.2
	M_OSWRIT	2,X'AF'	H.REXS,94	7.2
Program Hold Request	M.HOLD	1,X'58'	H.REXS,25	6.2
	M_HOLD	1,X'58'	H.REXS,25	7.2
Put User Context	M_PUTCTX	2,X'71'	H.EXEC,42	7.2
Read Descriptor	M.LOC	2,X'2C'	H.VOMM,13	6.2
	M_READD	2,X'2C'	H.VOMM,13	7.2
Read Record	M.READ	1,X'31'	H.IOCS,3	6.2
	M_READ	1,X'31'	H.IOCS,3	7.2
Read/Write Authorization File	N/A	N/A	H.VOMM,25	6.3
Receive Message Link Address	M.RCVR	1,X'6B'	H.REXS,43	6.2
	M_RCVR	1,X'6B'	H.REXS,43	7.2
Reconstruct Pathname	M.PNAM	2,X'2F'	H.VOMM,16	6.2
	M_CONSTRUCTPATH	2,X'2F'	H.VOMM,16	7.2
Reformat RRS Entry	M.NEWRRS	2,X'54'	H.REXS,76	6.2
Reinstate Privilege Mode to Privilege Task	M.PRIV	2,X'57'	H.REXS,78	6.2
	M_PRIVMODE	2,X'57'	H.REXS,78	7.2
Release Channel Reservation	M.RRES	1,X'3B'	H.IOCS,13	6.2
	M_RRES	1,X'3B'	H.IOCS,13	7.2
Release Dual-ported Disc/Set Dual-channel ACM Mode	M.RELP	1,X'27'	H.IOCS,27	6.2
	M_RELP	1,X'27'	H.IOCS,27	7.2
Release Exclusive File Lock	M.FXLR	1,X'22'	H.FISE,23	6.4
Release Exclusive Resource Lock	M.UNLOCK	2,X'45'	H.REMM,24	6.2
	M_UNLOCK	2,X'45'	H.REMM,24	7.2

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Release FHD Port	N/A	1,X'27'	H.IOCS,27	6.3
Release FHD Port-Base Mode	N/A	1,X'27'	H.IOCS,27	7.3
Release Synchronization File Lock	M.FSLR	1,X'24'	H.FISE,25	6.4
Release Synchronous Resource Lock	M.UNSYNC	2,X'47'	H.REMM,26	6.2
	M_UNSYNC	2,X'47'	H.REMM,26	7.2
Rename File	M.RENAM	2,X'2D'	H.VOMM,14	6.2
	M_RENAME	2,X'2D'	H.VOMM,14	7.2
Replace Permanent File	M.REPLAC	2,X'30'	H.VOMM,23	6.2
	M_REPLACE	2,X'30'	H.VOMM,23	7.2
Reserve Channel	M.RSRV	1,X'3A'	H.IOCS,12	6.2
	M_RSRV	1,X'3A'	H.IOCS,12	7.2
Reserve Dual-ported Disc/Set Single-channel ACM Mode	M.RESP	1,X'26'	H.IOCS,24	6.2
	M_RESP	1,X'26'	H.IOCS,24	7.2
Reserve FHD Port	N/A	1,X'26'	H.IOCS,24	6.3
Reserve FHD Port-Base Mode	N/A	1,X'26'	H.IOCS,24	7.3
Reserved for Interactive Debugger	N/A	2,X'56'	H.REXS,30	N/A
Reserved for Rapid File Allocation:	N/A			N/A
Zero MDT		2,X'AA'	H.MDT,1	
Locate/Read MDT Entry		2,X'AB'	H.MDT,2	
Update/Create MDT Entry		2,X'AC'	H.MDT,3	
Delete MDT Entry		2,X'AD'	H.MDT,4	
Reset Option Lower	M.ROPL	2,X'78'	H.TSM,14	6.2
	M_ROPL	2,X'78'	H.TSM,14	7.2
Resource Inquiry	M.INQUIRY	2,X'48'	H.REMM,27	6.2
	M_INQUIRER	2,X'48'	H.REMM,27	7.2
Resourcemark Lock	M.RSML	1,X'19'	H.REXS,62	6.2
	M_RSML	1,X'19'	H.REXS,62	7.2
Resourcemark Unlock	M.RSMU	1,X'1A'	H.REXS,63	6.2
	M_RSMU	1,X'1A'	H.REXS,63	7.2

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Resume Task Execution	M.SUME	1,X'53'	H.REXS,16	6.2
	M_SUME	1,X'53'	H.REXS,16	7.2
Rewind File	M.RWND	1,X'37'	H.IOCS,2	6.2
	M_REWIND	1,X'37'	H.IOCS,2	7.2
Rewrite Descriptor	M.REWRIT	2,X'2B'	H.VOMM,12	6.2
	M_REWRIT	2,X'2B'	H.VOMM,12	7.2
Rewrite Descriptor User Area	M.REWRTU	2,X'32'	H.VOMM,27	6.2
	M_REWRTU	2,X'32'	H.VOMM,27	7.2
Scan Terminal Input Buffer	M.TSCAN	1,X'5B'	H.TSM,2	6.2
	M_TSCAN	1,X'5B'	H.TSM,2	7.2
Send Message to Specified Task	M.SMSGR	1,X'6C'	H.REXS,44	6.2
	M_SMSGR	1,X'6C'	H.REXS,44	7.2
Send Run Request to Specified Task	M.SRUNR	1,X'6D'	H.REXS,45	6.2
	M_SRUNR	1,X'6D'	H.REXS,45	7.2
Set Asynchronous Task Interrupt	M.ASYNCH	1,X'1C'	H.REXS,68	6.2
	M_ASYNCH	1,X'1C'	H.REXS,68	7.2
Set Exception Handler	M_SETEXA	2,X'5C'	H.REXS,83	7.2
Set Exception Return Address	M_SETERA	2,X'79'	H.REXS,81	7.2
Set Exclusive File Lock	M.FXLS	1,X'21'	H.FISE,22	6.4
Set Exclusive Resource Lock	M.LOCK	2,X'44'	H.REMM,23	6.2
	M_LOCK	2,X'44'	H.REMM,23	7.2
Set IPU Bias	M.IPUBS	2,X'5B'	H.REXS,82	6.2
	M_IPUBS	2,X'5B'	H.REXS,82	7.2
Set Option Lower	M.SOPL	2,X'77'	H.TSM,13	6.2
	M_SOPL	2,X'77'	H.TSM,13	7.2
Set Synchronization File Lock	M.FSLS	1,X'23'	H.FISE,24	6.4
Set Synchronous Resource Lock	M.SETSYNC	2,X'46'	H.REMM,25	6.2
	M_SETSYNC	2,X'46'	H.REMM,25	7.2
Set Synchronous Task Interrupt	M.SYNCH	1,X'1B'	H.REXS,67	6.2
	M_SYNCH	1,X'1B'	H.REXS,67	7.2
Set Tabs in UDT	N/A	1,X'59'	H.TSM,5	N/A

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Set User Abort	M.SUAR	1,X'60'	H.REXS,26	6.2
Receiver Address	M_SUAR	1,X'60'	H.REXS,26	7.2
Set User Status	M.SETS	1,X'48'	H.REXS,7	6.2
Word	M_SETS	1,X'48'	H.REXS,7	7.2
Share Memory with Another Task	M.SHARE	1,X'71'	H.ALOC,12	6.4
Submit Job from Disc File	M.CDJS	1,X'61'	H.MONS,27	6.4
Suspend/Resume	M.SURE	5,X'00'	N/A	6.2
	M_SURE	5,X'00'	N/A	7.2
Suspend Task	M.SUSP	1,X'54'	H.REXS,17	6.2
Execution	M_SUSP	1,X'54'	H.REXS,17	7.2
System Console Type	M.TYPE	1,X'3F'	H.IOCS,14	6.2
	M_TYPE	1,X'3F'	H.IOCS,14	7.2
System Console Wait	M.CWAT	1,X'3D'	H.IOCS,26	6.2
	M_CWAT	1,X'3D'	H.IOCS,26	7.2
Task CPU Execution	M.XTIME	1,X'2D'	H.REXS,65	6.2
Time	M_XTIME	1,X'2D'	H.REXS,65	7.2
Task Option	M.PGOD	2,X'C0'	H.REXS,95	6.2
Doubleword Inquiry	M_OPTIONDWORD	2,X'C0'	H.REXS,95	7.2
Task Option Word	M.PGOW	1,X'4C'	H.REXS,24	6.2
Inquiry	M_OPTIONWORD	1,X'4C'	H.REXS,24	7.2
Terminate Task	M.EXIT	1,X'55'	H.REXS,18	6.2
Execution	M_EXIT	1,X'55'	H.REXS,18	7.2
Test Timer Entry	M.TSTT	1,X'46'	H.REXS,5	6.2
	M_TSTT	1,X'46'	H.REXS,5	7.2
Test User Status	M.TSTS	1,X'49'	H.REXS,8	6.2
Word	M_TSTS	1,X'49'	H.REXS,8	7.2
Time-of-Day Inquiry	M.TDAY	1,X'4E'	H.REXS,11	6.2
	M_TDAY	1,X'4E'	H.REXS,11	7.2
Trap On-line User's	M.TBRKON	1,X'5C'	H.TSM,6	6.2
Task	M_TBRKON	1,X'5C'	H.TSM,6	7.2
Truncate File	M.TRNC	2,X'26'	H.VOMM,7	6.2
	M_TRUNCATE	2,X'26'	H.VOMM,7	7.2
TSM Procedure	M.TSMPC	2,X'AE'	H.TSM,17	6.2
Call	M_TSMPC	2,X'AE'	H.TSM,17	7.2
TSM Task Detach	N/A	1,X'20'	H.TSM,3	N/A
Unlock and Dequeue Shared Memory	M.SMULK	1,X'1F'	H.ALOC,19	6.4

Description	Macro	SVC	Module, E.P.	Volume I Ref.Manual Section
Upspace	M.UPSP	1,X'10'	H.IOCS,20	6.2
	M_UPSP	1,X'10'	H.IOCS,20	7.2
User Name Specification	M.USER	1,X'74'	H.MONS,34	6.4
Validate Address Range	M.VADDR	2,X'59'	H.REXS,33	6.2
	M_VADDR	2,X'59'	H.REXS,33	7.2
Wait for Any No-wait Operation Complete, Message Interrupt, or Break Interrupt	M.ANYW	1,X'7C'	H.REXS,37	6.2
	M_ANYWAIT	1,X'7C'	H.REXS,37	7.2
Wait I/O	M.WAIT	1,X'3C'	H.IOCS,25	6.2
	M_WAIT	1,X'3C'	H.IOCS,25	7.2
Write EOF	M.WEOF	1,X'38'	H.IOCS,5	6.2
	M_WRITEEOF	1,X'38'	H.IOCS,5	7.2
Write Record	M.WRIT	1,X'32'	H.IOCS,4	6.2
	M_WRITE	1,X'32'	H.IOCS,4	7.2

B.3 SVC Listing

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
00-0A	Reserved			
0B	Reserved for Vector Processor			
0C	Reserved			
0D	Eject/Purge Routine	H.IOCS,22	N/A	6.3
	Eject/Purge Routine - Base Mode	H.IOCS,22	N/A	7.3
0E	Get Real Physical Address	H.REXS,90	M.RADDR M_RADDR	6.2 7.2
0F	Reserved for Vector Processor		N/A	N/A
10	Upspace	H.IOCS,20	M.UPSP M_UPSP	6.2 7.2
11	Reserved			
12	Disable User Break Interrupt	H.REXS,73	M.DSUB M_DSUB	6.2 7.2
13	Enable User Break Interrupt	H.REXS,72	M.ENUB M_ENUB	6.2 7.2
14	Get Device Mnemonic or Type Code	H.REXS,71	M.DEVID M_DEVID	6.2 7.2
15	Date and Time Inquiry	H.REXS,70	M.DATE M_DATE	6.2 7.2
16	ADI Maximum IOCBs	N/A	M.ADIMAX	N/A
17	ADI I/O	N/A	M.ADIO	N/A
18	ADI EAI	N/A	M.ADIEAI	N/A
19	Resourcemark Lock	H.REXS,62	M.RSML M_RSML	6.2 7.2
1A	Resourcemark Unlock	H.REXS,63	M.RSMU M_RSMU	6.2 7.2
1B	Set Synchronous Task Interrupt	H.REXS,67	M.SYNCH M_SYNCH	6.2 7.2
1C	Set Asynchronous Task Interrupt	H.REXS,68	M.ASYNCH M_ASYNCH	6.2 7.2

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
1D	End Action Wait	H.EXEC,40	M.EAWAIT M_AWAITACTION	6.2 7.2
1E	Activate Program at Given Time of Day	H.REXS,66	M.TURNON M_TURNON	6.2 7.2
1F	Unlock and Dequeue Shared Memory	H.ALOC,19	M.SMULK	6.4
20	TSM Task Detach	H.TSM,3	N/A	N/A
21	Set Exclusive File Lock	H.FISE,22	M.FXLS	6.4
22	Release Exclusive File Lock	H.FISE,23	M.FXLR	6.4
23	Set Synchronization File Lock	H.FISE,24	M.FSLS	6.4
24	Release Synchronization File Lock	H.FISE,25	M.FSLR	6.4
25	Execute Channel Program	H.IOCS,10	N/A	6.3
	Execute Channel Program - Base Mode	H.IOCS,10	N/A	7.3
26	ReserveFHD Port Reserve FHD Port - Base Mode	H.IOCS,24	N/A N/A	6.3 7.3
	Reserve Dual-ported Disc/Set Single-channel ACM Mode		M.RESP M_RESP	6.2 7.2
27	Release FHD Port Release FHD Port - Base Mode	H.IOCS,27	N/A N/A	6.3 7.3
	Release Dual-ported Disc/Set Dual-channel ACM Mode		M.RELP M_RELP	6.2 7.2
28	Convert ASCII Decimal to Binary	H.TSM,7	M.CONADB M_CONADB	6.2 7.2
29	Convert ASCII Hex to Binary	H.TSM,8	M.CONAHB M_CONAHB	6.2 7.2

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
2A	Convert Binary to ASCII Decimal	H.TSM,9	M.CONBAD M_CONBAD	6.2 7.2
2B	Convert Binary to ASCII Hex	H.TSM,10	M.CONBAH M_CONBAH	6.2 7.2
2C	No-wait I/O End-action Return	H.IOCS,34	M.XIEA	6.2
2D	Task CPU Execution Time	H.REXS,65	M.XTIME M_XTIME	6.2 7.2
2E	Disable Message Task Interrupt	H.REXS,57	M.DSMI M_DSMI	6.2 7.2
2F	Enable Message Task Interrupt	H.REXS,58	M.ENMI M_ENMI	6.2 7.2
30	Open File	H.IOCS,1	M.FILE	6.4
31	Read Record	H.IOCS,3	M.READ M_READ	6.2 7.2
32	Write Record	H.IOCS,4	M.WRIT M_WRITE	6.2 7.2
33	Advance Record	H.IOCS,7	M.FWRD M_ADVANCE	6.2 7.2
34	Advance File	H.IOCS,8	M.FWRD M_ADVANCE	6.2 7.2
35	Backspace Record	H.IOCS,9	M.BACK M_BACKSPACE	6.2 7.2
36	Backspace File	H.IOCS,19	M.BACK M_BACKSPACE	6.2 7.2
37	Rewind File	H.IOCS,2	M.RWND M_REWIND	6.2 7.2
38	Write EOF	H.IOCS,5	M.WEOF M_WRITEEOF	6.2 7.2
39	Close File	H.IOCS,23	M.CLSE M_CLSE	6.2 7.2
3A	Reserve Channel	H.IOCS,12	M.RSRV M_RSRV	6.2 7.2
3B	Release Channel Reservation	H.IOCS,13	M.RRES M_RRES	6.2 7.2

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
3C	Wait I/O	H.IOCS,25	M.WAIT M_WAIT	6.2 7.2
3D	System Console Wait	H.IOCS,26	M.CWAT M_CWAT	6.2 7.2
3E	Erase or Punch Trailer	H.IOCS,21	N/A	6.3
	Erase or Punch Trailer - Base Mode	H.IOCS,21	N/A	7.3
3F	System Console Type	H.IOCS,14	M.TYPE M_TYPE	6.2 7.2
40	Allocate File or Peripheral Device	H.MONS,21	M.ALOC	6.4
41	Deallocate File or Peripheral Device	H.MONS,22	M.DALC	6.4
42	Physical Device Inquiry	H.MONS,1	M.PDEV	6.4
43	Permanent File Address Inquiry	H.MONS,2	M.FADD	6.4
44	Memory Address Inquiry	H.REXS,3	M.ADRS M_ADRS	6.2 7.2
45	Create Timer Entry	H.REXS,4	M.SETT M_SETT	6.2 7.2
46	Test Timer Entry	H.REXS,5	M.TSTT M_TSTT	6.2 7.2
47	Delete Timer Entry	H.REXS,6	M.DLTT M_DLTT	6.2 7.2
48	Set User Status Word	H.REXS,7	M.SETS M_SETS	6.2 7.2
49	Test User Status Word	H.REXS,8	M.TSTS M_TSTS	6.2 7.2
4A	Change Priority Level	H.REXS,9	M.PRIL M_PRIL	6.2 7.2
4B	Connect Task to Interrupt	H.REXS,10	M.CONN M_CONN	6.2 7.2
4C	Task Option Word Inquiry	H.REXS,24	M.PGOW M_OPTIONWORD	6.2 7.2

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
4D	Arithmetic Exception Inquiry	H.REXS,23	M.TSTE M_TSTE	6.2 7.2
4E	Time-of-Day Inquiry	H.REXS,11	M.TDAY M_TDAY	6.2 7.2
4F	Memory Dump Request	H.REXS,12	M.DUMP M_DUMP	6.2 7.2
50	Load Overlay Segment	H.REXS,13	M.OLAY	6.2
51	Load and Execute Overlay	H.REXS,14	M.OLAY	6.2
52	Activate Task	H.REXS,15	M.ACTV M_ACTV	6.2 7.2
53	Resume Task Execution	H.REXS,16	M.SUME M_SUME	6.2 7.2
54	Suspend Task Execution	H.REXS,17	M.SUSP M_SUSP	6.2 7.2
55	Terminate Task Execution	H.REXS,18	M.EXIT M_EXIT	6.2 7.2
56	Abort Specified Task	H.REXS,19	M.BORT M_BORT	6.2 7.2
57	Abort Self	H.REXS,20	M.BORT M_BORT	6.2 7.2
58	Program Hold Request	H.REXS,25	M.HOLD M_HOLD	6.2 7.2
59	Set Tabs in UDT	H.TSM,5	N/A	N/A
5A	Delete Task	H.REXS,31	M.DELTSK M_DELTSK	6.2 7.2
5B	Scan Terminal Input Buffer	H.TSM,2	M.TSCAN M_TSCAN	6.2 7.2
5C	Trap On-line User's Task	H.TSM,6	M.TBRKON M_TBRKON	6.2 7.2
5D	Disconnect Task from Interrupt	H.REXS,38	M.DISCON M_DISCON	6.2 7.2
5E	Exit from Message Receiver	H.REXS,39	M.XMSGR	6.2
5F	Parameter Task Activation	H.REXS,40	M.PTSK M_PTSK	6.2 7.2

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
60	Set User Abort Receiver Address	H.REXS,26	M.SUAR M_SUAR	6.2 7.2
61	Submit Job from Disc File	H.MONS,27	M.CDJS	6.4
62	Abort With Extended Message	H.REXS,28	M.BORT M_BORT	6.2 7.2
63	Load and Execute Interactive Debugger	H.REXS,29	M.DEBUG M_DEBUG	6.2 7.2
64	Get Task Number	H.REXS,32	M.ID M_ID M.MYID M_MYID	6.2 7.2 6.2 7.2
65	Get Address Limits	H.REXS,41	M.GADRL	6.2
66	Debug Link Service	H.REXS,42	N/A	6.3
	Debug Link Service - Base Mode	H.REXS,42	N/A	7.3
67	Get Dynamic Task Execution Space	H.REMM,10	M.GE	6.2
68	Free Dynamic Task Execution Space	H.REMM,11	M.FE	6.2
69	Get Dynamic Extended Data Space	H.REMM,8	M.GD	6.2
6A	Free Dynamic Extended Indexed Data Space	H.REMM,9	M.FD	6.2
6B	Receive Message Link Address	H.REXS,43	M.RCVR M_RCVR	6.2 7.2
6C	Send Message to Specified Task	H.REXS,44	M.SMSGR M_SMSGR	6.2 7.2
6D	Send Run Request to Specified Task	H.REXS,45	M.SRUNR M_SRUNR	6.2 7.2
6E	Break/Task Interrupt Link/Unlink	H.REXS,46	M.BRK M_BRK	6.2 7.2
6F	Activate Task Interrupt	H.REXS,47	M.INT M_INT	6.2 7.2

SVC 1,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
70	Exit from Task Interrupt Level	H.REXS,48	M.BRKXIT M.XBRKR	6.2 6.2
71	Share Memory with Another Task	H.ALOC,12	M.SHARE	6.4
72	Get Shared Memory	H.ALOC,13	M.INCL	6.4
73	Permanent File Log	H.MONS,33	M.LOG	6.4
74	User Name Specification	H.MONS,34	M.USER	6.4
75	Create Permanent File	H.FISE,12	M.CREATE	6.4
76	Change Temporary File to Permanent	H.FISE,13	M.PERM	6.4
77	Delete Permanent File or Non-SYSGEN Memory Partition	H.FISE,14	M.DELETE	6.4
78	Reserved			
79	Free Shared Memory	H.ALOC,14	M.EXCL	6.4
7A	Get Message Parameters	H.REXS,35	M.GMSGP M_GMSGP	6.2 7.2
7B	Get Run Parameters	H.REXS,36	M.GRUNP M_GRUNP	6.2 7.2
7C	Wait for Any No-wait Operation Complete, Message Interrupt, or Break Interrupt	H.REXS,37	M.ANYW M_ANYWAIT	6.2 7.2
7D	Exit Run Receiver	H.REXS,49	M.XRUNR	6.2
7E	Exit from Message End-action Routine	H.REXS,50	M.XMEA	6.2
7F	Exit from Run Request End-action Routine	H.REXS,51	M.XREA	6.2
80-FFF	Available for customer use			

SVC 2,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
00-1F	Reserved			
20	Create Permanent File	H.VOMM,1	M.CPERM M_CREATEP	6.2 7.2
21	Create Temporary File	H.VOMM,2	M.TEMP M_CREATET	6.2 7.2
22	Create Memory Partition	H.VOMM,3	M.MEM M_MEM	6.2 7.2
23	Create Directory	H.VOMM,4	M.DIR M_DIR	6.2 7.2
24	Delete Resource	H.VOMM,5	M.DELR M_DELETER	6.2 7.2
25	Extend File	H.VOMM,6	M.EXTD M_EXTENDFILE	6.2 7.2
26	Truncate File	H.VOMM,7	M.TRNC M_TRUNCATE	6.2 7.2
27	Change Defaults	H.VOMM,8	M.DEFT M_DEFT	6.2 7.2
28	Change Temporary File to Permanent File	H.VOMM,9	M.TEMPER M_TEMPFILETOPERM	6.2 7.2
29	Log Resource or Directory	H.VOMM,10	M.LOGR M_LOGR	6.2 7.2
2A	Modify Descriptor	H.VOMM,11	M.MOD M_MOD	6.2 7.2
2B	Rewrite Descriptor	H.VOMM,12	M.REWRIT M_REWRIT	6.2 7.2
2C	Read Descriptor	H.VOMM,13	M.LOC M_READD	6.2 7.2
2D	Rename File	H.VOMM,14	M.RENAM M_RENAME	6.2 7.2
2E	Convert Pathname to Pathname Block	H.VOMM,15	M.PNAMB M_PNAMB	6.2 7.2
2F	Reconstruct Pathname	H.VOMM,16	M.PNAM M_CONSTRUCTPATH	6.2 7.2
30	Replace Permanent File	H.VOMM,23	M.REPLAC M_REPLACE	6.2 7.2
31	Modify Descriptor User Area	H.VOMM,26	M.MODU M_MODU	6.2 7.2

SVC 2,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
32	Rewrite Descriptor User Area	H.VOMM,27	M.REWRTU M_REWRTU	6.2 7.2
33	DBX Interface to H.PTRAC	N/A	N/A	N/A
34	Reserved for H.PTRAC			
35-3F	Reserved			
40	Include Memory Partition	H.REMM,12	M.INCLUDE	6.2
	Include Shared Image		M _INCLUDE	7.2
41	Exclude Memory Partition	H.REMM,14	M.EXCLUDE	6.2
	Exclude Shared Image		M _EXCLUDE	7.2
42	Open Resource	H.REMM,21	M.OPENR M_OPENR	6.2 7.2
43	Close Resource	H.REMM,22	M.CLOSER M_CLOSER	6.2 7.2
44	Set Exclusive Resource Lock	H.REMM,23	M.LOCK M_LOCK	6.2 7.2
45	Release Exclusive Resource Lock	H.REMM,24	M.UNLOCK M_UNLOCK	6.2 7.2
46	Set Synchronous Resource Lock	H.REMM,25	M.SETSYNC M_SETSYNC	6.2 7.2
47	Release Synchronous Resource Lock	H.REMM,26	M.UNSYNC M_UNSYNC	6.2 7.2
48	Resource Inquiry	H.REMM,27	M.INQUIRY M_INQUIRER	6.2 7.2
49	Mount Volume	H.REMM,17	M.MOUNT M_MOUNT	6.2 7.2
4A	Dismount Volume	H.REMM,19	M.DMOUNT M_DISMOUNT	6.2 7.2
4B	Get Memory in Byte Increments	H.REMM,28	M.MEMB M_GETMEMBYTES	6.2 7.2

SVC 2,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
4C	Free Memory in Byte Increments	H.REMM,29	M.MEMFRE M_FREEMEMBYTES	6.2 7.2
4D-4E	Reserved			
4F	Reserved			
50	Acquire Current Date/Time in ASCII Format	H.REXS,74	M.QATIM M_QATIM	6.2 7.2
	Acquire Current Date/Time in Binary Format	H.REXS,74	M.BTIM M_BTIM	6.2 7.2
	Acquire Current Date/Time in Byte Binary Format	H.REXS,74	M.BBTIM M_BBTIM	6.2 7.2
	Acquire System Date/Time in Any Format	H.REXS,74	M.GTIM M_GTIM	6.2 7.2
	Get Current Date and Time	H.REXS,74	M_GETTIME	7.2
51	Convert ASCII Date/Time to Byte Binary Format	H.REXS,75	M.CONABB M_CONABB	6.2 7.2
	Convert ASCII Date/Time to Standard Binary	H.REXS,75	M.CONASB M_CONASB	6.2 7.2
	Convert Binary Date/Time to ASCII Format	H.REXS,75	M.CONBAF M_CONBAF	6.2 7.2
	Convert Binary Date/Time to Byte Binary	H.REXS,75	M.CONBBY M_CONBBY	6.2 7.2
	Convert Byte Binary Date/Time to ASCII	H.REXS,75	M.CONBBA M_CONBBA	6.2 7.2
	Convert Byte Binary Date/Time to Binary	H.REXS,75	M.CONBYB M_CONBYB	6.2 7.2
	Convert System Date/Time Format	H.REXS,75	M.CTIM M_CTIM	6.2 7.2
	Convert Time	H.REXS,75	M_CONVERTTIME	7.2
52	Assign and Allocate Resource	H.REXS,21	M.ASSN M_ASSIGN	6.2 7.2

SVC 2,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
53	Deassign and Deallocate Resource	H.REXS,22	M.DASN M_DEASSIGN	6.2 7.2
54	Reformat RRS Entry	H.REXS,76	M.NEWRRS	6.2
55	Batch Job Entry	H.REXS,27	M.BATCH M_BATCH	6.2 7.2
56	Reserved for Interactive Debugger	H.REXS,30	N/A	N/A
57	Reinstate Privilege Mode to Privilege Task	H.REXS,78	M.PRIV M_PRIVMODE	6.2 7.2
58	Change Task to Unprivileged Mode	H.REXS,79	M.UPRIV M_UNPRIVMODE	6.2 7.2
59	Validate Address Range	H.REXS,33	M.VADDR M_VADDR	6.2 7.2
5A	Reserved			
5B	Set IPU Bias	H.REXS,82	M.IPUBS M_IPUBS	6.2 7.2
5C	Set Exception Handler	H.REXS,83	M_SETEXA	7.2
5D	Get Base Mode Task Address Limits	H.REXS,84	M_LIMITS	7.2
5E	Get Task Environment	H.REXS,85	M.ENVRMT M_ENVRMT	6.2 7.2
5F	Exit With Status	H.REXS,86	M_EXTSTS	7.2
60	Reserved			
61	Get Command Line	H.REXS,88	M.CMD M_CMD	6.2 7.2
62	Move Data to User Address	H.REXS,89	M.MOVE M_MOVE	6.2 7.2
63-6F	Reserved			
70	Get User Context	H.EXEC,41	M_GETCTX	7.2
71	Put User Context	H.EXEC,42	M_PUTCTX	7.2
72-74	Reserved for Symbolic Debugger/X32			
75	Reserved for MPX-32			

SVC 2,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
76	Allocate Shadow Memory	H.SHAD	N/A	N/A
77	Set Option Lower	H.TSM,13	M.SOPL M_SOPL	6.2 7.2
78	Reset Option Lower	H.TSM,14	M.ROPL M_ROPL	6.2 7.2
79	Set Exception Return Address	H.REXS,81	M_SETERA	7.2
7A	Get Terminal Function Definition	H.TSM,15	M.GETDEF M_GETDEF	6.2 7.2
7B	Get Address Limits	H.REXS,80	M.GADRL2	6.2
7C	Get Dynamic Extended Discontiguous Data Space	H.MEMM,9	M.GDD	6.2
7D	Get TSA Start Address	H.REXS,91	M.GTSAD M_GTSAD	6.2 7.2
7E	Physical Memory Read	H.REXS,93	M.OSREAD M_OSREAD	6.2 7.2
7F	Get Extended Memory Array	H.MEMM,14	N/A	6.3
	Get Extended Memory Array - Base Mode	H.MEMM,14	N/A	7.3
80-9F	Reserved for ACX-32			
A0-A3	Reserved for Swapper			
A4-A9	Reserved for Ada			
AA-AD	Reserved for Rapid File Allocation: Zero MDT Locate/Read MDT Entry Update/Create MDT Entry Delete MDT Entry	 H.MDT,1 H.MDT,2 H.MDT,3 H.MDT,4	N/A	N/A
AE	TSM Procedure Call	H.TSM,17	M.TSMPC M_TSMPC	6.2 7.2
AF	Physical Memory Write	H.REXS,94	M.OSWRIT M_OSWRIT	6.2 7.2
B0-BE	Reserved for RMSS			
BF	Reserved			

SVC 2,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
C0	Task Option Doubleword Inquiry	H.REXS,95	M.PGOD M_OPTIONDWORD	6.2 7.2
C1-C7	Reserved			
N/A	Allocate File Space	H.VOMM,19	N/A	6.3
N/A	Allocate Resource Descriptor	H.VOMM,17	N/A	6.3
N/A	Create File Control Block	N/A N/A	M.DFCB M_CREATEFCB	5.9.1 7.2
N/A	Create Temporary File	H.VOMM,24	N/A	6.3
N/A	Deallocate File Space	H.VOMM,20	N/A	6.3
N/A	Deallocate Resource Descriptor	H.VOMM,18	N/A	6.3
N/A	Execute Channel Program File Control Block	N/A	M_CHANPROGFCB	7.2
N/A	Read/Write Authorization File	H.VOMM,25	N/A	6.3

SVC 5,X'*nn*'	Description	Module, E.P.	Macro	Volume I Ref.Manual Section
00	Suspend/Resume	N/A	M.SURE M_SURE	6.2 7.2

C MPX-32 Abort and Crash Codes

C.1 AC – Accounting

AC01	INSUFFICIENT SLO SPACE FOR ACCOUNTING LISTING

C.2 AD – Address Specification Trap Handler (H.IP0C)

AD01	ADDRESS SPECIFICATION ERROR OCCURRED WITHIN THE OPERATING SYSTEM
AD02	ADDRESS SPECIFICATION ERROR OCCURRED WITHIN THE CURRENT TASK
AD03	TRAP OCCURRED WHILE NO TASKS WERE IN ACTIVE STATE
AD04	TRAP OCCURRED WITHIN ANOTHER INTERRUPT TRAP ROUTINE

C.3 AL – Allocator (H.ALOC) (Compatibility Mode Only)

AL01-AL06	Reserved
AL07	THE COMBINED FILE ASSIGNMENTS FOR A TASK EXCEEDS NUMBER SPECIFIED. THE CATALOGED ASSIGNMENTS ARE COMBINED WITH THOSE DEFINED BY $ASSIGN STATEMENTS. SEE CATALOGER FILES DIRECTIVE AND RECATALOG IF NEEDED.
AL08	AN ASSIGNED PERMANENT FILE IS NONEXISTENT
AL09	AN ASSIGNED DEVICE IS NOT CONFIGURED IN THE SYSTEM. AN ASSIGNED DEVICE IS OFF-LINE.
AL10-AL11	Reserved
AL12	UNABLE TO LOAD PROGRAM BECAUSE OF I/O ERROR OR ADDRESSING INCONSISTENCIES IN LOAD MODULE PREAMBLE
AL13	AN UNRECOVERABLE I/O ERROR HAS OCCURRED DURING THE READ OF THE TASK PREAMBLE INTO THE TSA
AL14	Reserved
AL15	AN ASSIGNED DEVICE TYPE IS NOT CONFIGURED IN THE SYSTEM
AL16	A RESIDENT REQUEST HAS BEEN ISSUED FOR A TASK REQUIRING AN SLO, SBO, SGO OR SYC FILE. RESIDENT TASKS CANNOT USE SYSTEM FILES.
AL17-AL18	Reserved

AL19	A FILE CODE TO FILE CODE ASSIGNMENT (ASSIGN4) HAS BEEN MADE TO AN UNDEFINED FILE CODE. A FILE CODE MUST BE DEFINED BEFORE A SECOND FILE CODE CAN BE EQUATED BY AN ASSIGN4.
AL20	USER ATTEMPTED DEALLOCATION OF TSA
AL21	DESTROYED TASK MIDL WAS DETECTED WHILE ATTEMPTING TO ALLOCATE DYNAMIC EXECUTION SPACE
AL22	A SOFTWARE CHECKSUM ERROR HAS OCCURRED DURING TASK LOADING
AL23	AN INVALID USER NAME IS CATALOGED WITH THE TASK. THE USER NAME IS NOT CONTAINED IN THE M.KEY FILE OR A VALID KEY IS NOT SPECIFIED.
AL24	ACCESS TO AN ASSIGNED PERMANENT FILE IS BY PASSWORD ONLY, AND A VALID PASSWORD WAS NOT INCLUDED ON THE CATALOGED ASSIGNMENT OR JOB CONTROL STATEMENT ASSIGNMENT
AL25	UNDEFINED RESOURCE REQUIREMENT SUMMARY (RRS) TYPE (INTERNAL FORMAT OF AN ASSIGNMENT STATEMENT IS WRONG)
AL26	THE TASK HAS REQUESTED MORE BLOCKING BUFFERS THAN WERE SPECIFIED DURING CATALOG. SEE CATALOGER BUFFER DIRECTIVE AND RECATALOG IF NEEDED.
AL27	THERE ARE NO FREE ENTRIES IN SHARED MEMORY TABLE FOR GLOBAL, DATAPOOL, CSECT, OR OTHER SHARED AREAS
AL28	TASK IS ATTEMPTING TO SHARE AN UNDEFINED GLOBAL OR DATAPOOL MEMORY PARTITION
AL29	TASK IS ATTEMPTING TO EXCLUDE UNDEFINED MEMORY PARTITION
AL30	THE REQUESTED DEVICE IS ALREADY ASSIGNED TO THE REQUESTING TASK VIA ANOTHER FILE CODE. USE ASSIGN4 OR DEALLOCATE BEFORE REALLOCATING.
AL31	LOGICAL FILE CODE ALREADY ALLOCATED BY CALLER (E.G., A CARD READER MAY BE ASSIGNED TO LFC 'IN' AND A MAGNETIC TAPE CANNOT BE ASSIGNED TO THE SAME FILE CODE). USE ASSIGN4 OR DEALLOCATE BEFORE REALLOCATING.
AL32	DYNAMIC COMMON BLOCK MAY NOT BE ASSIGNED VIA ASSIGN1 DIRECTIVE
AL33	SHARED MEMORY DEFINITION CONFLICTS WITH CALLER'S ADDRESS SPACE
AL34	SHARED MEMORY PARTITION NOT DEFINED IN DIRECTORY
AL35	ATTEMPT TO SHARE A DIRECTORY ENTRY THAT IS NOT A MEMORY PARTITION

AL36	INVALID PASSWORD SPECIFIED FOR SHARED MEMORY PARTITION
AL37	ATTEMPT TO EXCLUDE UNDEFINED SHARED MEMORY PARTITION
AL38	ATTEMPT TO ACTIVATE A PRIVILEGED TASK BY UNAUTHORIZED OWNER
AL39	SHARED MEMORY ENTRY NOT FOUND
AL40	PARTITION DEFINITION NOT FOUND IN DIRECTORY
AL41	DIRECTORY DEFINITION NOT A DYNAMIC PARTITION
AL42	INVALID PASSWORD FOR A MEMORY PARTITION
AL43	TASK HAS ATTEMPTED TO ALLOCATE AN UNSHARED RESOURCE THAT WAS NOT AVAILABLE DURING TASK ACTIVATION IN A MEMORY-ONLY ENVIRONMENT
AL44	UNABLE TO RESUME 'SYSBUILD' TASK DURING INITIAL TASK ACTIVATION IN A MEMORY-ONLY ENVIRONMENT
AL45	UNABLE TO DEALLOCATE INPUT DEVICE AFTER DYNAMIC TASK ACTIVATION IN A MEMORY-ONLY ENVIRONMENT
AL46	TASK HAS ATTEMPTED TO SHARE MEMORY VIA A DYNAMIC MEMORY PARTITION IN A MEMORY-ONLY ENVIRONMENT
AL47	DYNAMIC MEMORY PARTITIONS CANNOT BE GREATER THAN 1 MEGABYTE
AL48	THE USER HAS ATTEMPTED TO EXCLUDE A SHARED PARTITION WHOSE ASSOCIATED MAP BLOCKS ARE NOT DESIGNATED AS BEING SHARED IN THE TASK'S TSA
AL49	THE TASK'S DSECT SPACE REQUIREMENTS OVERLAP THE TASK'S TSA SPACE REQUIREMENTS
AL50	THE TASK'S DSECT SPACE REQUIREMENTS OVERLAP THE TASK'S CSECT SPACE REQUIREMENTS, OR IF NO CSECT, LOAD MODULE IS TOO LARGE TO FIT IN USER'S ADDRESS SPACE
AL51	DESTROYED TASK MIDL DETECTED WHILE ATTEMPTING TO ALLOCATE SYSTEM BUFFER SPACE
AL52	AN ERROR CONDITION PERTAINING TO FILE SYSTEM STRUCTURES HAS OCCURRED. THIS ERROR IS NOT A FUNCTION OF THE COMPATIBILITY INTERFACE.
AL53	DESTROYED TASK MIDL WAS DETECTED WHILE ATTEMPTING TO ALLOCATE EXTENDED INDEXED DATA SPACE
AL54	INVALID COMPATIBLE RRS TYPE
AL55	ACCESS MODE IS NOT ALLOWED

C.4 AT – ANSI Labeled Tapes

AT01	INCORRECT OR NO RUN PARAMETERS RECEIVED
AT02	INCORRECT STATUS RETURNED FROM J.ATAPE RUN REQUEST
AT03	AN ERROR OCCURRED
AT04	I/O ERROR OCCURRED ON TAPE

C.5 AU – Auto-Start Trap Processor

AU01	TRAP OCCURRED ON AUTO-START

C.6 BT – Block Mode Timeout Trap

BT01	BLOCK MODE TIMEOUT TRAP

C.7 CM – Call Monitor Interrupt Processor (H.IP27 and H.IP0A)

CM01	CALL MONITOR INTERRUPT PROCESSOR CANNOT LOCATE THE 'CALM' INSTRUCTION
CM02	EXPECTED 'CALM' INSTRUCTION DOES NOT HAVE CALM (X'30') OPCODE
CM03	INVALID 'CALM' NUMBER
CM04	'CALM' NUMBER TOO LOW (OUT OF BOUNDS)
CM05	'CALM' NUMBER TOO BIG (OUT OF BOUNDS)

C.8 CP – Cache

CP01	CACHE PARITY ERROR OCCURRED WITHIN THE OPERATING SYSTEM
CP02	CACHE PARITY ERROR OCCURRED IN TASK BODY
CP03	TRAP OCCURRED WHILE NO TASKS WERE IN ACTIVE STATE
CP04	TRAP OCCURRED IN ANOTHER INTERRUPT TRAP ROUTINE

C.9 EX – Exit/Abort

EX01	AN ABORT HAS OCCURRED IN THE TASK EXIT SEQUENCE
EX02	AN ABORT HAS OCCURRED DURING THE TASK ABORT SEQUENCE AND HAS BEEN CHANGED TO A DELETE (KILL) TASK SEQUENCE
EX03	USER ATTEMPTED TO GO TO AN ANY WAIT STATE FROM AN END-ACTION ROUTINE

C.10 FS – File System (H.MONS)(Compatibility Mode Only)

FS01	UNRECOVERABLE I/O ERROR TO THE DIRECTORY
FS02	UNRECOVERABLE I/O ERROR TO FILE SPACE ALLOCATION MAP
FS03	ATTEMPT TO ADD A NEW FILE, BUT THE DIRECTORY IS FULL
FS04	A DISC ALLOCATION MAP CHECKSUM ERROR WAS DETECTED
FS05	ATTEMPT TO ALLOCATE DISC SPACE THAT IS ALREADY ALLOCATED
FS06	ATTEMPT TO DEALLOCATE DISC SPACE THAT IS NOT ALLOCATED
FS07	USER HAS CALLED AN ENTRY POINT IN H.FISE THAT NO LONGER EXISTS

C.11 HE – Online Help Facility

HE01	ABNORMAL TERMINATION WHILE TRANSLATING HELP FILES (HELPT)

C.12 HT – Halt Trap Processor (H.IPHT)

HT01	AN ATTEMPT WAS MADE TO EXECUTE A HALT INSTRUCTION IN USER'S PROGRAM
HT02	AN ATTEMPT WAS MADE TO EXECUTE A HALT INSTRUCTION IN AN INTERRUPT TRAP ROUTINE
HT03	AN ATTEMPT WAS MADE TO EXECUTE A HALT INSTRUCTION WHEN NO TASKS WERE IN AN ACTIVE STATE
HT04	Reserved
HT05	AN ATTEMPT WAS MADE TO EXECUTE A HALT INSTRUCTION WHEN USER WAS UNMAPPED

C.13 IO – Input/Output Control Supervisor (H.IOCS)

IO01	Reserved
IO02	AN UNPRIVILEGED TASK IS ATTEMPTING TO READ OR WRITE DATA INTO AN UNMAPPED ADDRESS
IO03	AN UNPRIVILEGED TASK IS ATTEMPTING TO READ DATA INTO PROTECTED MEMORY
IO04-IO05	Reserved
IO06	INVALID BLOCKING BUFFER CONTROL CELLS IN BLOCKED FILE ENCOUNTERED. PROBABLE CAUSES: (1) FILE IS IMPROPERLY BLOCKED, (2) BLOCKING BUFFER IS DESTROYED, OR (3) TRANSFER ERROR DURING FILE INPUT.
IO07	THE TASK HAS ATTEMPTED TO PERFORM AN OPERATION WHICH IS NOT VALID FOR THE DEVICE TO WHICH THE USER'S FILE IS ASSIGNED(E.G., A READ OPERATION SPECIFIED FOR A FILE ASSIGNED TO THE LINE PRINTER).
IO08	DEVICE ASSIGNMENT IS REQUIRED FOR AN UNPRIVILEGED TASK TO USE THIS SERVICE
IO09	ILLEGAL OPERATION ON THE SYC FILE
IO10-IO14	Reserved
IO15	A TASK HAS REQUESTED A TYPE OPERATION AND THE TYPE CONTROL PARAMETER BLOCK(TCPB) SPECIFIED INDICATES THAT AN OPERATION ASSOCIATED WITH THAT TCPB IS ALREADY IN PROGRESS
IO16	INVALID BLOCKING BUFFER CONTROL CELL(S) ENCOUNTERED DURING WRITE OF BLOCKED FILE. THIS ERROR IS USUALLY CAUSED BY A USER SPECIFIED BLOCKING BUFFER THAT HAS BEEN DESTROYED.
IO17	OPEN ATTEMPTED ON A FILE AND FPT HAS NO MATCHING FILE CODE. PROBABLE CAUSE: (1) BAD OR MISSING RRS IN PREAMBLE (2) LFC IN FCB HAS BEEN DESTROYED.
IO18	Reserved
IO19	AN ERROR HAS OCCURRED IN THE REMM CLOSE PROCEDURE
IO20	AN ERROR HAS OCCURRED IN THE REMM OPEN PROCEDURE
IO21	IOCS HAS ENCOUNTERED AN UNRECOVERABLE I/O ERROR IN ATTEMPTING TO PROCESS AN I/O REQUEST ON BEHALF OF A TASK
IO22	AN ILLEGAL IOCS ENTRY POINT HAS BEEN ENTERED BY A TASK
IO23	A H.VOMM DENIAL HAS OCCURRED IN READING THE RESOURCE DESCRIPTOR TO GET MORE SEGMENT DEFINITIONS

IO24	ILLEGAL ADDRESS, TRANSFER COUNT OR TRANSFER TYPE (I.E., IMPROPER BOUNDING FOR DATA TYPE) SPECIFIED IN THE FCB
IO25-IO27	Reserved
IO28	ILLEGAL OPERATION ATTEMPTED ON AN OUTPUT ACTIVE FILE OR DEVICE
IO29	Reserved
IO30	ILLEGAL OR UNEXPECTED VOLUME NUMBER OR REEL ID ENCOUNTERED ON MAGNETIC TAPE
IO31	Reserved
IO32	CALLING TASK HAS ATTEMPTED TO PERFORM A SECOND READ ON A '$' STATEMENT THROUGH THE SYC FILE
IO33	READ WITH BYTE GRANULARITY REQUEST MADE WITH NEGATIVE BYTE OFFSET
IO34	READ WITH BYTE GRANULARITY REQUEST MADE WITHOUT SETTING RANDOM ACCESS BIT IN FCB
IO35	READ WITH BYTE GRANULARITY REQUESTS ARE VALID FOR UNBLOCKED FILES ONLY
IO36-IO37	Reserved
IO38	WRITE ATTEMPTED ON UNIT OPENED IN READ-ONLY MODE. A READ-WRITE OPEN WILL BE FORCED TO READ-ONLY IF TASK HAS ONLY READ ACCESS TO UNIT.
IO39	Reserved
IO40	INVALID TRANSFER COUNT. TRANSFER COUNT TOO LARGE FOR TRANSFER TYPE, TRANSFER COUNT NOT AN EVEN MULTIPLE OF TRANSFER TYPE, OR DATA ADDRESS NOT BOUNDED FOR TRANSFER TYPE.
IO41	BLOCKING ERROR DURING NON-DEVICE ACCESS
IO42	BLOCKED DATA MANAGEMENT MODULE (H.BKDM) IS NOT CONFIGURED IN THE SYSTEM
IO43	INPUT/OUTPUT CONTROL LIST (IOCL) OR DATA ADDRESS NOT IN CONTIGUOUS 'E' MEMORY (GPMC DEVICES ONLY)
IO44	NON-DEVICE ACCESS I/O ERROR. THIS ERROR MAY BE THE RESULT OF CHANNEL/CONTROLLER INITIALIZATION FAILURE.
IO45	MULTIVOLUME MAGNETIC TAPE MODULE (H.MVMT) IS NOT CONFIGURED IN THE SYSTEM
IO46	Reserved
IO47	CLASS 'E' DEVICE TCW IS NOT IN CLASS 'E' MEMORY. THIS TYPE OF ERROR INDICATES A MAP FAILURE.

IO48-IO49	Reserved
IO50	AN UNPRIVILEGED USER ATTEMPTED TO EXECUTE A PHYSICAL CHANNEL PROGRAM
IO51	A 'TESTSTAR' COMMAND WAS USED IN A LOGICAL CHANNEL PROGRAM
IO52	A LOGICAL CHANNEL WAS TOO LARGE TO BE MOVED TO MEMORY POOL
IO53	A 'TIC' COMMAND FOLLOWS A 'TIC' COMMAND IN A LOGICAL CHANNEL PROGRAM
IO54	A 'TIC' COMMAND ATTEMPTED TO TRANSFER TO AN ADDRESS WHICH IS NOT WORD BOUNDED
IO55	ILLEGAL ADDRESS IN LOGICAL IOCL. ADDRESS IS NOT IN USER'S LOGICAL ADDRESS SPACE.
IO56	A READ-BACKWARD COMMAND WAS USED IN A LOGICAL CHANNEL PROGRAM
IO57	ILLEGAL IOCL ADDRESS. IOCL MUST BE LOCATED IN THE FIRST 128K WORDS OF MEMORY.
IO58-IO60	Reserved
IO61	INVALID LFC IN FCB
IO62	ERROR OCCURRED ON IMPLICIT OPEN
IO63-IO76	Reserved
IO77	ATTEMPT TO USE DATA FLOW CONTROL (OTHER THAN WISM), THAT IS NOT SUPPORTED BY THE CURRENTLY INSTALLED CONTROLLER
IO78	ATTEMPT TO ISSUE AN EXECUTE CHANNEL PROGRAM TO A WRITE SUB-CHANNEL AND THE SUB-CHANNEL WAS NOT IN DUAL CHANNEL MODE
IO79	Reserved
IO80	ILLEGAL ACCESS MODE FOR VOLUME RESOURCE
IO81-IO97	Reserved
IO98	H.VOMM DENIAL HAS OCCURRED ON IOCS AUTOMATIC FILE EXTENSION REQUEST FOR THE LFC SPECIFIED IN THE ABORT MESSAGE
IO99	INTERNAL SYSTEM ERROR DETECTED AT THE ADDRESS RELATIVE TO IOCS WHICH IS SPECIFIED IN THE ABORT MESSAGE

C.14 IP – IPU

IP01	ABNORMAL TASK TERMINATION IN IPU

C.15 LD – Task Activation Loading (H.TAMM)

LD01	LOAD CODE SECTION ERROR
LD02	CODE SECTION CHECKSUM ERROR
LD03	BIAS CODE ERROR
LD04	CODE MATRIX CHECKSUM ERROR
LD05	LOAD DATA SECTION ERROR
LD06	DATA SECTION CHECKSUM ERROR
LD07	BIAS DATA ERROR
LD08	DATA MATRIX CHECKSUM ERROR
LD09	GCF R/O RELOCATION ERROR
LD10	GCF R/W RELOCATION ERROR

C.16 MC – Machine Check Trap

MC01	MACHINE CHECK TRAP

C.17 MF – Map Fault Trap

MF01	A MAP FAULT TRAP HAS OCCURRED. THIS IS THE RESULT OF A BAD MEMORY REFERENCE OUTSIDE OF THE USER'S ADDRESSABLE SPACE.

C.18 MM – Memory Disk

MM01	REQUEST FOR MEMORY DISC I/O TO A LOCATION OUTSIDE THE MEMORY DISC BOUNDARIES

C.19 MP – Memory Parity Trap (H.IP02)

MP01	MEMORY ERROR OCCURRED IN A TASK'S LOGICAL ADDRESS SPACE. THIS IS AN INTERNAL OR CPU FAILURE. RERUN TASK.
MP02	MEMORY ERROR OCCURRED IN ANOTHER INTERRUPT TRAP ROUTINE (NESTED TRAPS, CONTEXT LOST)
MP03	MEMORY ERROR OCCURRED WHILE NO TASKS WERE IN THE ACTIVE STATE
MP04	MEMORY ERROR OCCURRED IN A MAP BLOCK RESERVED FOR THE O/S
MP05	ERROR OCCURRED WHILE CURRENT TASK WAS IN THE UNMAPPED MODE

C.20 MS – System Services (H.MONS) (Compatibility Mode Only)

MS01	PERMANENT FILE ADDRESS INQUIRY SERVICE FOUND A NUMBER OF ALLOCATION UNITS IN THE UNIT DEFINITION TABLE THAT DO NOT CORRESPOND TO ANY KNOWN DISC.
MS02-MS08	Reserved
MS09	TASK HAS ATTEMPTED TO CONNECT A TASK TO AN INTERRUPT LEVEL NOT DEFINED FOR INDIRECTLY CONNECTED TASKS
MS10-MS11	Reserved
MS12	OVERLAY IS PASSWORD PROTECTED
MS13-MS15	Reserved
MS16	TASK HAS REQUESTED DYNAMIC ALLOCATION WITH AN INVALID FUNCTION CODE
MS17	FILE NAME CONTAINS CHARACTERS OUTSIDE RANGE OF X'20' TO X'5F', INCLUSIVELY
MS18-MS20	Reserved
MS21	MULTIVOLUME MAGNETIC TAPE ALLOCATION REQUEST MADE TO SCRATCH (SCRA) TAPE
MS22	MULTI-VOLUME MAGNETIC TAPE ALLOCATION REQUEST MADE ON SHARED TAPE DRIVE
MS23	TASK HAS ISSUED A 'MOUNT MESSAGE ONLY' ALLOCATION REQUEST TO A NON-ALLOCATED DRIVE OR TO A DEVICE WHICH IS NOT A MAGNETIC TAPE

MS24	TASK HAS SPECIFIED AN ILLEGAL VOLUME NUMBER (ZERO IF TAPE IS MULTIVOLUME, NONZERO IF TAPE IS SINGLE VOLUME)
MS25-MS27	Reserved
MS28	A PERMANENT FILE LOG HAS BEEN REQUESTED, BUT THE ADDRESS SPECIFIED FOR STORAGE OF THE DIRECTORY ENTRY IS NOT CONTAINED WITHIN THE CALLING TASK'S LOGICAL ADDRESS SPACE
MS29	Reserved
MS30	TASK HAS ATTEMPTED TO OBTAIN A PERMANENT FILE LOG IN A MEMORY-ONLY ENVIRONMENT
MS31	USER ATTEMPTED TO GO TO THE ANY-WAIT STATE FROM AN END-ACTION ROUTINE
MS32	Reserved
MS33	ALLOCATION ERROR IN RTM M.ALOC CALL
MS34-MS86	Reserved
MS87	NO DENIAL RETURN ADDRESS SPECIFIED ON CALM M.ALOC EMULATION

C.21 NM – Nonpresent Memory Trap

NM01	A NONPRESENT MEMORY TRAP ERROR CONDITION HAS OCCURRED.

C.22 OC – Operator Communications

OC01	THE OPERATOR HAS REQUESTED THAT THE TASK BE ABORTED

C.23 PT – Task Activation (J.TSM)

PT01	INVALID ATTEMPT TO MULTICOPY A UNIQUE TASK
PT02	FILE SPECIFIED IS NOT IN DIRECTORY
PT03	UNABLE TO ALLOCATE FILE
PT04	FILE IS NOT A VALID LOAD MODULE OR EXECUTABLE IMAGE
PT05	DQE IS NOT AVAILABLE
PT06	READ ERROR ON RESOURCE DESCRIPTOR

PT07	READ ERROR ON LOAD MODULE
PT08	INSUFFICIENT LOGICAL/PHYSICAL ADDRESS SPACE FOR TASK ACTIVATION
PT09	CALLING TASK IS UNPRIVILEGED
PT10	INVALID PRIORITY
PT11	INVALID SEND BUFFER ADDRESS OR SIZE
PT12	INVALID RETURN BUFFER ADDRESS OR SIZE
PT13	INVALID NO-WAIT MODE END ACTION ROUTINE ADDRESS
PT14	MEMORY POOL UNAVAILABLE
PT15	DESTINATION TASK RECEIVER QUEUE FULL
PT16	INVALID PSB ADDRESS
PT17	RRS LIST EXCEEDS 384 WORDS
PT18	INVALID RRS ENTRY IN PARAMETER BLOCK

C.24 PV – Privilege Violation Trap

PV01	PRIVILEGE VIOLATION TRAP

C.25 RC – Record Manager

RC01	LESS THAN ONE BLOCK ON READ
RC02	NOT A MULTIPLE NUMBER OF BLOCKS READ
RC03	NO MORE IOC'S AVAILABLE
RC04	ERROR CONDITION ON READ
RC05	PREMATURE END-OF-FILE
RC06	END-OF-MEDIUM ON OUTPUT FILE
RC07	WRITE ATTEMPTED ON UNOPENED FILE
RC08	USER RECORD SIZE TOO LARGE
RC09	READ NOT ALLOWED AFTER WRITE
RC10	ERROR ON WRITE
RC11	END-OF-MEDIUM ON OUTPUT FILE

RC12	INTERNAL FILE POSITION ERROR
RC13	RESOURCE CANNOT BE OPENED
RC14	INTERNAL FILE POSITION ERROR
RC15	INVALID BLOCKING BUFFER CELL

C.26 RE – Restart

RE01	RESTART IS INVALID IN BATCH OR COMMAND FILE MODE

C.27 RF – Rapid File Allocation

RF01	INVALID PATHNAME
RF02	PATHNAME CONSISTS OF VOLUME ONLY
RF03	VOLUME NOT MOUNTED
RF04	Reserved
RF05	FILE IS NOT A PERMANENT FILE
RF06	Reserved
RF07	RESOURCE DOES NOT EXIST
RF08	RESOURCE NAME IN USE
RF09	Reserved
RF10	MDT ENTRY UNAVAILABLE
RF11-RF14	Reserved
RF15	VOLUME MUST BE MOUNTED PUBLIC
RF16-RF59	Reserved
RF60	INVALID MODE
RF61-RF98	Reserved
RF99	WARNING, INPUT ERRORS ENCOUNTERED, CHECK SLO OUTPUT

C.28 RM – Resource Management (H.REMM)

RM01	UNABLE TO LOCATE RESOURCE
RM02	ACCESS MODE NOT ALLOWED

RM03	TOO MANY ASSIGNMENTS
RM04	BLOCKING BUFFER SPACE NOT AVAILABLE OR INVALID BUFFER ADDRESS
RM05	SHARED MEMORY TABLE (SMT) ENTRY NOT FOUND
RM06	TOO MANY MOUNT REQUESTS
RM07	STATIC ASSIGN TO DYNAMIC COMMON
RM08	UNRECOVERABLE I/O ERROR
RM09	INVALID USAGE SPECIFICATION
RM10	INVALID PARAMETER ADDRESS
RM11	INVALID RESOURCE REQUIREMENT SUMMARY (RRS) ENTRY
RM12	INVALID LFC TO LFC ASSIGNMENT
RM13	DEVICE NOT IN SYSTEM OR OFF-LINE
RM14	RESOURCE ALREADY ALLOCATED BY TASK
RM15	INVALID SYC/SGO ASSIGNMENT
RM16	COMMON CONFLICTS WITH TASK ADDRESS SPACE
RM17	DUPLICATE LFC ASSIGNMENT
RM18	INVALID DEVICE SPECIFICATION
RM19	INVALID RESOURCE ID (RID)
RM20	VOLUME UNASSIGNED OR ACCESS NOT ALLOWED
RM21	UNABLE TO MOUNT. J.MOUNT RUN REQUEST FAILED
RM22	RESOURCE MARKED FOR DELETION
RM23	ASSIGNED DEVICE IS MARKED OFF-LINE
RM24	UNABLE TO LOCATE MOUNTED VOLUME TABLE(MVT) ENTRY
RM25	RANDOM ACCESS NOT ALLOWED
RM26	ATTEMPT TO WRITE ON SYC
RM27	RESOURCE ALREADY OPENED IN DIFFERENT MODE
RM28	INVALID ACCESS SPECIFICATION AT OPEN
RM29	INVALID FILE CONTROL BLOCK(FCB) ADDRESS OR UNASSIGNED LFC IN FCB
RM30	INVALID ALLOCATION INDEX

RM31	RESOURCE NOT OPEN
RM32	LOCK NOT OWNED BY THIS TASK
RM33	RESOURCE IS NOT ALLOCATED IN A SHARABLE MODE
RM34	SYSTEM ADMINISTRATOR ATTRIBUTE IS REQUIRED TO MOUNT A PUBLIC VOLUME
RM35	RESOURCE IS NOT A SHARED IMAGE
RM36	PHYSICAL MEMORY ALREADY ALLOCATED
RM37	ATTEMPT TO ALLOCATE NONPRESENT PHYSICAL MEMORY
RM38	TIME OUT WAITING FOR RESOURCE
RM39	UNABLE TO PERFORM WRITE BACK
RM40	INVALID LOAD MODULE
RM41	INVALID PHYSICAL ADDRESS SPECIFIED
RM42	USER REQUESTED ABORT OF MOUNT PROCESS
RM43	USER REQUESTED HOLD ON MOUNT PROCESS
RM44	WRITEBACK REQUESTED AND SHARED IMAGE HAS NO WRITEBACK SECTION
RM45	LOADING ERROR DURING INCLUSION OF READ ONLY SECTION OF SHARED IMAGE
RM46	UNABLE TO OBTAIN RESOURCE DESCRIPTOR LOCK (MULTIPORT ONLY)
RM47	LOADING ERROR DURING INCLUSION OF READ/WRITE SECTION OF SHARED IMAGE
RM48	INCOMPATIBLE LOAD ADDRESSES FOR SHARED IMAGE
RM49	TASK HAS REQUESTED EXCESSIVE NUMBER OF MULTICOPIED SHARED IMAGES WITH NO READ ONLY SECTION
RM50	RESOURCE IS LOCKED BY ANOTHER TASK
RM51	SHAREABLE RESOURCE IS ALLOCATED BY ANOTHER TASK IN AN INCOMPATIBLE ACCESS MODE
RM52	VOLUME SPACE IS NOT AVAILABLE
RM53	ASSIGNED DEVICE IS NOT AVAILABLE
RM54	UNABLE TO ALLOCATE RESOURCE FOR SPECIFIED USAGE
RM55	ALLOCATED RESOURCE TABLE (ART) SPACE IS NOT AVAILABLE

RM56	TASK REQUIRES SHADOW MEMORY AND NONE IS CONFIGURED
RM57	VOLUME IS NOT AVAILABLE FOR MOUNT WITH REQUESTED USAGE
RM58	SHARED MEMORY TABLE (SMT) SPACE IS NOT AVAILABLE
RM59	MOUNTED VOLUME TABLE (MVT) SPACE IS NOT AVAILABLE
RM60	RESOURCE DESCRIPTOR SPACE DEFINITION CONFLICT
RM61	UNABLE TO LOCATE OR RETRIEVE RESOURCE DESCRIPTOR
RM62	INVALID OPTION IN CNP
RM63	SEGMENTED TASK SUPPORT NOT PRESENT.
RM64	THE TASK'S DSECT SPACE REQUIREMENTS OVERLAP THE TASK'S TASK SERVICE AREA(TSA) SPACE REQUIREMENTS
RM65	THE TASK'S DSECT SPACE REQUIREMENTS OVERLAP THE TASK'S CSECT SPACE REQUIREMENTS, OR IF NO CSECT, LOAD MODULE IS TOO LARGE TO FIT IN USER'S ADDRESS SPACE
RM66	SOFTWARE CHECKSUM. ERROR MAY BE FIXED BY RECATALOGING.
RM67	EXCESSIVE MEMORY REQUEST
RM68	EXCESSIVE VOLUME SPACE REQUESTED
RM69	INVALID USERNAME SPECIFIED
RM70	INVALID PRIVILEGED ACTIVATION
RM71	Reserved
RM72	UNABLE TO RESUME SYSINIT ON TAPE ACTIVATION
RM73	FILE OVERLAP HAS OCCURRED. PLEASE CHECK THE SYSTEM CONSOLE
RM74	LOADING ERROR
RM75	INVALID WORK VOLUME/DIRECTORY
RM76	USER ATTEMPTED DEALLOCATION OF TSA
RM77	A TASK HAS DESTROYED THE ALLOCATION LINKAGES IN ITS DYNAMIC EXPANSION SPACE
RM78	UNABLE TO LOAD TASK DEBUGGER WITH TASK
RM79	INVALID CALLER NOTIFICATION PACKET (CNP) ADDRESS
RM80	SHARED IMAGE VERSION LEVEL IS NOT COMPATIBLE WITH EXECUTABLE IMAGE

RM81	INVALID ACTIVATION OF A BASE MODE TASK ON A SYSTEM CONFIGURED FOR NON-BASE TASK EXECUTION.
RM82	INVALID ACTIVATION OF AN ADA TASK ON A SYSTEM CONFIGURED WITHOUT ADA SUPPORT.
RM83	INSUFFICIENT LOGICAL ADDRESS SPACE TO ACTIVATE TASK
RM84	INVALID LOGICAL POSITION FOR EXTENDED MPX
RM85	PTRACE DEBUG REQUESTED AND H.PTRAC NOT CONFIGURED
RM86	CANNOT DISMOUNT THE SYSTEM VOLUME.
RM87	PUBLIC VOLUME DISMOUNT DENIED DUE TO COMPATIBLE MODE PUBLIC DISMOUNT OPTION SET FOR THIS SYSTEM.
RM88	PUBLIC DISMOUNT DENIED. SYSTEM ADMINISTRATOR ATTRIBUTE REQUIRED FOR THIS OPERATION.
RM89	PUBLIC DISMOUNT DENIED DUE TO MISSING OPTION FOR PUBLIC VOLUME IN THE DISMOUNT REQUEST
RM90	GCL LOADMODULE OR SHIM CANNOT BE RELOCATABLE
RM91	UNABLE TO ACCESS VOLUME DUE TO PENDING PHYSICAL DISMOUNT.
RM92	READ ONLY OR READ WRITE LOAD ADDRESS IS INVALID
RM93	UNABLE TO PERFORM PHYSICAL MOUNT DUE TO SYSTEM SHUTDOWN IN PROGRESS.
RM94	J.MOUNT ATTEMPTED TO MOUNT AN UNFORMATTED DISC VOLUME.
RM95	AN UNBIASED TASK REQUIRES SHADOW MEMORY ON A SYSTEM WITH NO OVERLAPPING CPU/IPU SHADOW REGION
RM96	A BIASED TASK REQUIRES SHADOW MEMORY THAT DOES NOT EXIST ON THE SPECIFIED PROCESSOR
RM97	Reserved
RM98	THE TASK REQUIRES MORE SHADOW MEMORY THAN EXISTS

C.29 RX – Resident Executive Services (H.REXS)

RX01	Reserved
RX02	INVALID FUNCTION CODE SPECIFIED FOR REQUEST TO CREATE A TIMER ENTRY. VALID CODES ARE ACP (1), RSP OR RST (2), STB (3), RSB (4) AND RQI (5).
RX03	TASK ATTEMPTED TO SET/RESET A BIT OUTSIDE OF A STATIC PARTITION OR THE OPERATING SYSTEM.

RX04	THE REQUESTING TASK IS UNPRIVILEGED OR HAS ATTEMPTED TO CREATE A TIMER ENTRY TO REQUEST AN INTERRUPT WITH A PRIORITY LEVEL OUTSIDE THE RANGE OF X'12' TO X'7F', INCLUSIVELY
RX05	INVALID FUNCTION CODE HAS BEEN SPECIFIED FOR REQUEST TO SET USER STATUS WORD
RX06	UNPRIVILEGED TASK ATTEMPTED TO RESET A TASK PRIORITY LEVEL, OR A PRIVILEGED TASK ATTEMPTED TO RESET A TASK PRIORITY TO A LEVEL OUTSIDE THE RANGE OF 1 TO 64, INCLUSIVELY
RX07	CANNOT LOAD OVERLAY SEGMENT DUE TO SOFTWARE CHECKSUM OR DATA ERROR
RX08	OVERLAY IS NOT IN THE DIRECTORY
RX09	Reserved
RX10	OVERLAY HAS AN INVALID PREAMBLE
RX11	AN UNRECOVERABLE I/O ERROR HAS OCCURRED DURING OVERLAY LOADING
RX12	Reserved
RX13	FUNCTION CODE SUPPLIED TO A DATE/TIME SERVICE IS OUT OF RANGE
RX14	DESTINATION BUFFER ADDRESS IS INVALID OR PROTECTED
RX15	ATTEMPT TO SET EXCEPTION RETURN ADDRESS WHEN ARITHMETIC EXCEPTION NOT IN PROGRESS
RX16-RX24	Reserved
RX25	OPERATOR HAS ABORTED TASK IN RESPONSE TO MOUNT MESSAGE
RX26-RX28	Reserved
RX29	TASK HAS ATTEMPTED TO LOAD THE INTERACTIVE TASK DEBUGGER OVERLAY IN A MEMORY-ONLY ENVIRONMENT
RX30-RX31	Reserved
RX32	INVALID DQE ADDRESS
RX33	OVERLAY LINKAGES HAVE BEEN DESTROYED BY LOADING A LARGER OVERLAY
RX34	TASK HAS MADE A BREAK RECEIVER EXIT CALL WHILE NO BREAK IS ACTIVE
RX35	Reserved

RX36	STATUS IN REGISTER ZERO IS NOT A ZERO OR A VALID ABORT CODE
RX37-RX85	Reserved
RX86	TASK HAS MADE AN END ACTION ROUTINE EXIT WHILE END ACTION WAS NOT ACTIVE
RX87	Reserved
RX88	RESERVED FOR DEBUG LINK SERVICE
RX89	AN UNPRIVILEGED TASK HAS ATTEMPTED TO REESTABLISH AN ABORT RECEIVER (OTHER THAN M.IOEX)
RX90	TASK HAS MADE A RUN REQUEST END ACTION ROUTINE EXIT WHILE THE RUN REQUEST INTERRUPT WAS NOT ACTIVE
RX91	TASK HAS ATTEMPTED NORMAL EXIT WITH A TASK INTERRUPT STILL ACTIVE
RX92	TASK HAS ATTEMPTED NORMAL EXIT WITH MESSAGES IN ITS RECEIVER QUEUE
RX93	AN INVALID RECEIVER EXIT BLOCK (RXB) ADDRESS WAS ENCOUNTERED DURING MESSAGE EXIT
RX94	AN INVALID RECEIVER EXIT BLOCK (RXB) RETURN BUFFER ADDRESS WAS ENCOUNTERED DURING MESSAGE EXIT
RX95	TASK HAS MADE A MESSAGE EXIT WHILE THE MESSAGE INTERRUPT WAS NOT ACTIVE
RX96	AN INVALID RECEIVER EXIT BLOCK (RXB) ADDRESS WAS ENCOUNTERED DURING RUN RECEIVER EXIT
RX97	AN INVALID RECEIVER EXIT BLOCK (RXB) RETURN BUFFER ADDRESS WAS ENCOUNTERED DURING RUN RECEIVER EXIT
RX98	TASK HAS MADE A RUN RECEIVER EXIT WHILE THE RUN RECEIVER INTERRUPT WAS NOT ACTIVE
RX99	TASK HAS MADE A MESSAGE END-ACTION ROUTINE EXIT WHILE THE MESSAGE INTERRUPT WAS NOT ACTIVE

C.30 SB – System Binary Output

SB01	AN I/O ERROR HAS BEEN ENCOUNTERED ON THE DEVICE ASSIGNED AS THE SYSTEM BINARY (PUNCHED) OUTPUT DEVICE
SB02	THE SYSTEM OUTPUT PROGRAM HAS ENCOUNTERED AN UNRECOVERABLE I/O ERROR IN ATTEMPTING TO READ A PUNCHED OUTPUT FILE FROM DISC
SB03	DENIAL OF FILE CODE TO FILE CODE ALLOCATION FOR J.SOUT2 INDICATES LOSS OF SYSTEM INTEGRITY

SB04	SYSTEM BINARY OUTPUT ABORTED BY OPERATOR
SB05	NO TIMER ENTRY FOR SYSTEM BINARY OUTPUT (SYSTEM FAULT)
SB06	FIVE ECHO CHECK ERRORS DETECTED WHILE ATTEMPTING TO PUNCH A SINGLE CARD

C.31 SC – System Check Trap Processor

SC01	SYSTEM CHECK TRAP OCCURRED AT AN ADDRESS LOCATED WITHIN THE OPERATING SYSTEM
SC02	SYSTEM CHECK TRAP OCCURRED WITHIN THE CURRENT TASK'S SPACE
SC03	SYSTEM CHECK TRAP OCCURRED AT A TIME WHEN THERE WERE NO TASKS CURRENTLY BEING EXECUTED (C.PRNO EQUALS ZERO)
SC04	SYSTEM CHECK TRAP OCCURRED WITHIN ANOTHER TRAP (C.GINT DOES NOT EQUAL '1')

C.32 SD – SCSI Disk

SD00	NO ADDITIONAL SENSE INFORMATION
SD01	NO INDEX/SECTOR SIGNAL
SD02	NO SEEK COMPLETE
SD03	WRITE FAULT
SD04	DRIVE NOT READY
SD05	DRIVE NOT SELECTED
SD06	NO TRACK ZERO FOUND
SD07	MULTIPLE DRIVES SELECTED
SD08	LOGICAL UNIT COMMUNICATIONS FAILURE
SD09	TRACK FOLLOWING ERROR
SD10-SD15	Reserved
SD16	ID CRC OR ECC ERROR
SD17	UNRECOVERED READ ERROR OF DATA BLOCKS
SD18	NO ADDRESS MARK FOUND IN ID FIELD

SD19	NO ADDRESS MARK FOUND IN DATA FIELD
SD20	NO RECORD FOUND
SD21	SEEK POSITIONING ERROR
SD22	DATA SYNCHRONIZATION MARK ERROR
SD23	RECOVERED READ DATA WITH TARGET'S READ RETRIES (NOT WITH ECC)
SD24	RECOVERED READ DATA WITH TARGET'S ECC CORRECTION (NOT WITH RETRIES)
SD25	DEFECT LIST ERROR
SD26	PARAMETER OVERRUN
SD27	SYNCHRONOUS TRANSFER ERROR
SD28	PRIMARY DEFECT LIST NOT FOUND
SD29	COMPARE ERROR
SD30	RECOVERED ID WITH TARGET'S ECC CORRECTION
SD31	Reserved
SD32	INVALID COMMAND OPERATION CODE
SD33	ILLEGAL LOGICAL BLOCK ADDRESS. ADDRESS GREATER THAN THE LBA RETURNED BY THE READ CAPACITY DATA WITH PMI BIT NOT SET IN CDB
SD34	ILLEGAL FUNCTION FOR DEVICE TYPE
SD35	Reserved
SD36	ILLEGAL FIELD IN CDB
SD37	INVALID LUN
SD38	INVALID FIELD IN PARAMETER LIST
SD39	WRITE PROTECTED
SD40	MEDIUM CHANGE
SD41	POWER ON OR RESET OR BUS DEVICE RESET OCCURRED
SD42	MODE SELECT PARAMETERS CHANGED
SD43-SD47	Reserved
SD48	IMCOMPATIBLE CARTRIDGE

SD49	MEDIUM FORMAT CORRUPTED
SD50	NO DEFECT SPARE LOCATION AVAILABLE
SD51-SD63	Reserved
SD64	RAM FAILURE
SD65	DATA PATH DIAGNOSTIC FAILURE
SD66	POWER ON DIAGNOSTIC FAILURE
SD67	MESSAGE REJECT ERROR
SD68	INTERNAL CONTROLLER ERROR
SD69	SELECT/RESELECT FAILED
SD70	UNSUCCESSFUL SOFT RESET
SD71	SCSI INTERFACE PARITY ERROR
SD72	INITIATOR DETECTED ERROR
SD73	INAPPROPRIATE/ILLEGAL MESSAGE

C.33 SG – System Generator (SYSGEN)

SG01	INVALID LOADER FUNCTION CODE IN BINARY OBJECT MODULE FROM THE SYSTEM RESIDENT MODULE (OBJ) FILE
SG02	INVALID BINARY RECORD READ FROM SYSTEM RESIDENT MODULE (OBJ) FILE (BYTE 0 MUST BE X'FF' OR X'DF')
SG03	SEQUENCE ERROR IN MODULE BEING READ FROM TEMPORARY FILE
SG04	CHECKSUM ERROR IN MODULE BEING READ FROM TEMPORARY FILE
SG05	UNABLE TO FIND CDT AND/OR UDT FOR I/O MODULE LOAD
SG06	UNABLE TO OBTAIN ADDITIONAL MEMORY REQUIRED FOR RESIDENT SYSTEM IMAGE MODULE LOADING
SG07	UNABLE TO OBTAIN MEMORY REQUIRED FOR RESIDENT SYSTEM IMAGE CONSTRUCTION
SG08	NON-RELOCATABLE BYTE STRING ENCOUNTERED IN BINARY MODULE BEING PROCESSED FROM TEMPORARY FILE
SG09	UNABLE TO ALLOCATE TEMPORARY FILE SPACE
SG10	OVERRUN OF SYSGEN ADDRESS SPACE BY SYSTEM BEING GENERATED. PROBABLE ERRONEOUS SIZE SPECIFICATION IN PATCH OR POOL DIRECTIVE.

SG11	SEQUENCE ERROR WHILE READING OBJECT MODULE FROM FILE ASSIGNED TO 'OBJ'
SG12	CHECKSUM ERROR WHILE READING OBJECT MODULE FROM FILE ASSIGNED TO 'OBJ'
SG13	UNABLE TO ALLOCATE DISC SPACE FOR SYMTAB FILE. POSSIBLE CAUSES ARE INSUFFICIENT DISC SPACE OR ACCESS RIGHTS DENIAL.
SG14	UNABLE TO ALLOCATE DISC SPACE FOR SYSTEM IMAGE FILE. POSSIBLE CAUSES ARE INSUFFICIENT DISC SPACE, ACCESS RIGHTS DENIAL, OR ATTEMPTING TO SYSGEN OVER CURRENT DEFAULT IMAGE.
SG15	MAXIMUM NUMBER (240) OF SYMBOL TABLE/PATCH FILE ENTRIES EXCEEDED
SG16	MISSING SYSTEM OR SYMTAB DIRECTIVE
SG17	INVALID IPU INTERVAL TIMER PRIORITY. MUST NOT BE BETWEEN X'78' AND X'7F'.
SG18	MAXIMUM SIZE OF 88K FOR TARGET SYSTEM HAS BEEN EXCEEDED
SG19	ATTEMPT TO DEFINE INTERRUPT VECTORING ROUTINE AS SYSTEM REENTRANT. ONLY DEVICE HANDLERS MAY BE SYSTEM REENTRANT.
SG20	UNABLE TO FIND "LINK" DEVICE IN UDT
SG21	INSUFFICIENT ROOM IN MEMORY POOL FOR DOWNLOAD FILE LIST
SG22	Reserved
SG23	SHARE DIRECTIVE SPECIFIED WITHOUT ENOUGH SMT ENTRIES. ENTRIES MUST EXCEED OR BE EQUAL TO THE NUMBER OF PARTITIONS PLUS MEMORY DISCS.
SG24	ATTEMPT TO DEFINE PARTITION STARTING MAPBLOCK NUMBER IN OPERATING SYSTEM AREA
SG25	ATTEMPT TO DEFINE PARTITION STARTING MAPBLOCK NUMBER IN NON-CONFIGURED PHYSICAL MEMORY
SG26	ATTEMPT TO USE A MODULE INCOMPATIBLE WITH THE TARGET MACHINE TYPE. THE OFFENDING MODULE NAME IS THE LAST ENTRY ON THE LISTING FOLLOWED BY THREE ASTERISKS (***).
SG27	THE DEVICE SPECIFIED IN EITHER THE SWAPDEV, SID, LOD OR POD DIRECTIVE IS NOT INCLUDED IN THE CONFIGURATION BEING BUILT
SG28	THE NULL DEVICE SPECIFICATION WHICH IS REQUIRED TO BE INCLUDED IN EVERY CONFIGURATION IS MISSING

SG29	SYSINIT OBJECT MODULE MISSING ON SYSGEN OBJECT INPUT FILE (OBJ). IT MUST BE THE LAST MODULE.
SG30	THE FILE ASSIGNED TO FILE CODE OBJ DOES NOT CONTAIN VALID OBJECT CODE
SG31	THE GENERATED IMAGE CONTAINS UNSATISFIED EXTERNAL REFERENCES. SEE THE SLO OUTPUT FOR MORE DETAILS. THIS IS NOT A FATAL ABORT AND THE SYSTEM IMAGE IS PRODUCED.
SG32	ONE OR MORE REQUESTED OBJECT MODULES COULD NOT BE LOCATED ON THE INPUT OBJECT FILE. SEE THE SLO OUTPUT FOR MORE DETAILS. THIS IS NOT A FATAL ABORT AND THE SYSTEM IMAGE IS PRODUCED.
SG33	EVENT TRACE HAS BEEN ENABLED WITH NO MEMORY PARTITION RESERVED FROM X'78000' TO X'80000'
SG34	Reserved
SG35	INSUFFICIENT MEMORY POOL FOR STATIC PARTITION
SG36	UNMAPPED DEBUG MODULE (H.DBUG2) IS MISSING ON SYSGEN OBJECT INPUT FILE. IT MUST BE THE LAST MODULE IF THE SYSTEM DEBUGGER IS TO BE CONFIGURED.
SG37	COMMUNICATION REGION + DSECT + ADAPTIVE REGION EXCEEDS 16KW
SG38	MPX EXTENDED CODE AREA EXTENDS PAST LOGICAL LIMIT
SG39	INVALID MPX EXTENDED CODE AREA LOGICAL MAP START
SG40	DIRECTIVE ERRORS ENCOUNTERED. IMAGE PRODUCED.
SG41	H.IPPF COULD NOT BE LOCATED ON THE INPUT OBJECT FILE. MODULE IS NECESSARY FOR DEMAND PAGE.
SG42-SG97	Reserved
SG98	ERROR ENCOUNTERED DURING OBJECT PROCESSING PRECEDED BY MESSAGE DESCRIBING THE ERROR CONDITION
SG99	DIRECTIVE ERRORS ENCOUNTERED

C.34 SH – Shadow Memory (J.SHAD)

SH01	J.SHAD ABORTED. SEE OUTPUT (UT IF INTERACTIVE OR SLO IF BATCH), FOR ACTUAL ERROR DESCRIPTION(S).

C.35 SN – System Input Task (J.SSIN)

SN00	INVALID RUN REQUEST PARAMETERS

C.36 SS – Sort/Merge (FSORT2)

SS01	CTL NOT ALLOCATED
SS02	HEADER DIRECTIVE MISSING
SS03	CONTROL FILE EMPTY
SS04	DIRECTIVE CODE NOT VALID
SS05-SS06	Reserved
SS07	OUTPUT FILE CODE (OUT) NOT ALLOCATED
SS08	RECORD LENGTH NOT DIVISIBLE INTO INPUT PHYSICAL RECORD LENGTH
SS09	RECORD LENGTH EXCEEDS INPUT PHYSICAL RECORD LENGTH
SS10	INPUT RECORD LENGTH EXCEEDS MAXIMUM ALLOWED (4095)
SS11	RECORD LENGTH NOT DIVISIBLE INTO OUTPUT PHYSICAL RECORD LENGTH
SS12	RECORD LENGTH EXCEEDS OUTPUT BLOCK LENGTH
SS13	OUTPUT PHYSICAL RECORD LENGTH EXCEEDS MAXIMUM ALLOWED (4095)
SS14	..1 PRESENT BUT NOT A DISC FILE
SS15	..2 PRESENT BUT NOT A DISC FILE
SS16	COMPARISON INDICATOR NOT VALID
SS17	Reserved
SS18	WK1 HAS BEEN ALLOCATED BY THE USER
SS19	WK2 HAS BEEN ALLOCATED BY THE USER
SS20	FIELD DIRECTIVE ERROR: STARTING POSITION IS GREATER THAN FIELD ENDING POSITION
SS21	FIELD DIRECTIVE ERROR: STARTING POSITION EXCEEDS RECORD LENGTH
SS22	FIELD DIRECTIVE ERROR: ENDING POSITION EXCEEDS LOGICAL RECORD LENGTH
SS23-SS27	Reserved
SS28	INAPPROPRIATE COMBINATION OF TOURNAMENT PARAMETERS EXCEEDS MEMORY POOL LIMITS
SS29	DISC SPACE CANNOT BE ALLOCATED FOR WORK FILE 1

SS30	DISC SPACE CANNOT BE ALLOCATED FOR WORK FILE 2
SS31	FILE TO FILE ALLOCATION FOR WORKFILE HAS FAILED
SS32	SORT BUFFER TOO SMALL
SS33-SS39	Reserved
SS40	INPUT FILES ARE EMPTY: NO RECORD INPUT OR SORTED
SS41	WK1 OR WK2 FILES TOO SMALL
SS42	MERGE ONLY SELECTED BUT NO MERGE FILES (MG1-MG8) ARE ASSIGNED
SS43-SS47	Reserved
SS48	SORT ATTEMPTED WITHOUT GOOD CALL TO SORT:HDR
SS49-SS57	Reserved
SS58	INAPPROPRIATE COMBINATION OF BUFFER PARAMETERS DETECTED DURING OUTPUT PHASE
SS59	END OF MEDIUM DETECTED ON THE OUT FILE
SS60-SS68	Reserved
SS69	COMPARE TABLE TYPE DESTROYED: SORT PROBLEM
SS70-SS97	Reserved
SS98	ERROR OPENING FILE LO
SS99	ERROR OPENING FILE OUT

C.37 ST – System Output Task (J.SOUT)

ST01	UNRECOVERABLE WRITE ERROR TO DESTINATION DEVICE
ST02	UNABLE TO PERFORM ALLOCATION OF SEPARATOR FILE CODE
ST03	UNABLE TO ISSUE MAGNETIC TAPE MOUNT MESSAGE VIA ALLOCATION SERVICE

Whenever a system output task aborts, the task may be restarted with the OPCOM REPRINT or REPUNCH commands.

C.38 SV – SVC Trap Processor (H.IP06)

SV01	UNPRIVILEGED TASK ATTEMPTING TO USE M.CALL
SV02	INVALID SVC NUMBER

SV03	UNPRIVILEGED TASK ATTEMPTING TO USE A 'PRIVILEGED-ONLY' SERVICE
SV04	INVALID SVC TYPE
SV05	UNPRIVILEGED TASK ATTEMPTING TO USE M.RTRN
SV06	INVALID MODULE NUMBER OR ENTRY POINT
SV07	ATTEMPTING TO USE A SVC WHICH IS INVALID FOR BASE REGISTER OPERATIONS
SV08	SVC 0, 1 OR 2 ATTEMPTED THAT WOULD RESULT IN A TSA STACK OVERFLOW (I.E. T.REGP GREATER THAN T.LASTP)
SV09	ATTEMPT TO USE A COMPATIBLE MODE SERVICE WITH NOCMS SPECIFIED IN SYSGEN

C.39 SW – Swap Scheduler Task (J.SWAPR)

SW01	I/O ERROR ON INSWAP OR OUTSWAP
SW02	EOM DETECTED ON SWAP FILE
SW03	CAN NOT CREATE SWAP FILE SPACE DIRECTORY IN MEMORY POOL
SW04	SWAP FILE SPACE DIRECTORY IS FULL
SW05	TASK HAS REQUESTED INSWAP BUT WAS NEVER OUTSWAPPED

C.40 SX – System Output Executive (J.SOEX)

SX01	INVALID RUN REQUEST HEADCELL COUNT
SX02	LOAD MODULE J.SOUT DOES NOT EXIST

C.41 SY – System Initialization (SYSINIT)

SY01	SYSTEM HALT OCCURRED DURING SYSINIT PHASE ONE PROCESSING
SY02	SYSTEM HALT DUE TO MEMORY PARITY ERROR BEING DETECTED IN THE OPERATING SYSTEM

C.42 TD – Terminal Type Set/Reset Utility (J.TSET)

TD01	ATTEMPTED TO RUN J.TSET IN BATCH MODE
TD02	J.TSET WAS UNABLE TO OPEN UT FOR PROCESSING

C.43 TS – Terminal Support

TS01	USER REQUESTED REMOVAL FROM A BREAK REQUEST
TS02	USER REQUESTED REMOVAL FROM A RESOURCE WAIT STATE QUEUE
TS03	TASK RUNNING FROM SPECIFIED TERMINAL WAS ABORTED WHEN THE TERMINAL DISCONNECTED
TS04	REMOVAL OF A JOB WAS REQUESTED

C.44 UI – Undefined Instruction Trap

UI01	UNDEFINED INSTRUCTION TRAP
UI02	UNEXPECTED DEBUGX32 BREAKPOINT FOUND AND DEBUGX32 NOT ATTACHED

C.45 VF – Volume Formatter (J.VFMT)

VF01	ERROR HAS OCCURRED. SEE SLO FILE FOR EXPLANATION.
VF02	OPEN FAILURE ON AUDIT TRAIL DEVICE/FILE
VF03	EOF/EOM ON AUDIT TRAIL DEVICE/FILE
VF04	I/O ERROR ON AUDIT TRAIL DEVICE/FILE

C.46 VM – Volume Management Module (H.VOMM)

In some cases, H.VOMM displays H.REMM abort conditions. If a user calls an H.VOMM service which in turn calls an H.REMM service for processing and an abort condition occurs within the H.REMM processing, the abort condition is returned to H.VOMM which displays it to the user in the format `10`*xx* where *xx* is the specific H.REMM abort condition. For example, abort condition `1026` indicates H.REMM error 26 has occurred. The TSM $ERR command can be used to determine the reason for the error, i.e., `$ERR RM26`.

VM01	INVALID PATHNAME
VM02	PATHNAME CONSISTS OF VOLUME ONLY

VM03	VOLUME NOT MOUNTED
VM04	DIRECTORY DOES NOT EXIST
VM05	DIRECTORY NAME IN USE
VM06	DIRECTORY CREATION NOT ALLOWED AT SPECIFIED LEVEL
VM07	RESOURCE DOES NOT EXIST
VM08	RESOURCE ALREADY EXISTS
VM09	RESOURCE DESCRIPTOR UNAVAILABLE
VM10	DIRECTORY ENTRY UNAVAILABLE
VM11	REQUIRED FILE SPACE UNAVAILABLE
VM12	UNRECOVERABLE I/O ERROR READING DMAP
VM13	UNRECOVERABLE I/O ERROR WRITING DMAP
VM14	UNRECOVERABLE I/O ERROR READING RESOURCE DESCRIPTOR
VM15	UNRECOVERABLE I/O ERROR WRITING RESOURCE DESCRIPTOR
VM16	UNRECOVERABLE I/O ERROR READING SMAP
VM17	UNRECOVERABLE I/O ERROR WRITING SMAP
VM18	UNRECOVERABLE I/O ERROR READING DIRECTORY
VM19	UNRECOVERABLE I/O ERROR WRITING DIRECTORY
VM20	PROJECTGROUP NAME OR KEY INVALID
VM21	Reserved
VM22	INVALID FILE CONTROL BLOCK(FCB) OR LFC
VM23	PARAMETER ADDRESS SPECIFICATION ERROR
VM24	RESOURCE DESCRIPTOR NOT CURRENTLY ALLOCATED
VM25	PATHNAME BLOCK OVERFLOW
VM26	FILE SPACE NOT CURRENTLY ALLOCATED
VM27	'CHANGE DEFAULTS' NOT ALLOWED
VM28	RESOURCE CANNOT BE ACCESSED IN REQUESTED MODE OR DEFAULT SYSTEM IMAGE FILE CANNOT BE DELETED
VM29	OPERATION NOT ALLOWED ON THIS RESOURCE TYPE (RESOURCE IS NOT CORRECT TYPE)
VM30	REQUIRED PARAMETER WAS NOT SPECIFIED

VM31	FILE EXTENSION DENIED. SEGMENT DEFINITION AREA FULL.
VM32	FILE EXTENSION DENIED. FILE WOULD EXCEED MAXIMUM SIZE ALLOWED.
VM33	I/O ERROR OCCURRED WHEN RESOURCE WAS ZEROED
VM34	REPLACEMENT FILE CANNOT BE ALLOCATED
VM35	INVALID DIRECTORY ENTRY
VM36	DIRECTORY AND FILE ARE NOT ON THE SAME VOLUME
VM37	AN UNIMPLEMENTED ENTRY POINT HAS BEEN CALLED
VM38	REPLACEMENT FILE IS ALLOCATED BY ANOTHER TASK AND BIT 0 IN THE CNP OPTION FIELD IS NOT SET, OR FILE IS ALLOCATED BY OTHER CPU IN MULTI-PORT ENVIRONMENT
VM39	OUT OF SYSTEM SPACE
VM40	CANNOT ALLOCATE FAT/FPT WHEN CREATING A TEMPORARY FILE
VM41	DEALLOCATE ERROR IN ZEROING FILE
VM42	RESOURCE DESCRIPTOR DESTROYED OR THE RESOURCE DESCRIPTOR AND THE DIRECTORY ENTRY LINKAGE HAS BEEN DESTROYED
VM43	INVALID RESOURCE SPECIFICATION
VM44	INTERNAL LOGIC ERROR FROM RESOURCE MANAGEMENT MODULE (H.REMM). ABORT TASK, TRY A DIFFERENT TASK AND IF IT FAILS, REBOOT SYSTEM.
VM45	ATTEMPTED TO MODIFY MORE THAN ONE RESOURCE DESCRIPTOR AT THE SAME TIME OR ATTEMPTED TO REWRITE A RESOURCE DESCRIPTOR PRIOR TO MODIFYING IT
VM46	RESOURCE DESCRIPTOR IS LOCKED BY ANOTHER CPU (MULTI-PORT ONLY)
VM47	DIRECTORY CONTAINS ACTIVE ENTRIES AND CANNOT BE DELETED
VM48	A RESOURCE DESCRIPTOR'S LINK COUNT IS ZERO
VM49	ATTEMPTING TO DELETE A PERMANENT RESOURCE WITHOUT SPECIFYING A PATHNAME OR PATHNAME BLOCK VECTOR
VM50	RESOURCE DESCRIPTOR CONTAINS UNEXPECTED RESOURCE DESCRIPTOR TYPE
VM51	DIRECTORY ENTRY DELETED BUT FAILED TO RELEASE FILE SPACE

VM52	AN ATTEMPT WAS MADE TO DEALLOCATE FREE SPACE OR TO ALLOCATE SPACE THAT IS CURRENTLY ALLOCATED ON A VOLUME OTHER THAN SYSTEM DISC
VM53	THE FILE SPACE CREATED IS LESS THAN THE SPACE REQUESTED
VM54-VM98	Reserved
VM99	AN ATTEMPT WAS MADE TO DEALLOCATE FREE SPACE OR TO ALLOCATE SPACE THAT IS CURRENTLY ALLOCATED ON THE SYSTEM VOLUME

C.47 VO – Volume Manager (VOLMGR)

VO01	ERROR HAS OCCURRED. SEE SLO FILE FOR EXPLANATION.
VO02	OPEN FAILURE ON AUDIT TRAIL DEVICE/FILE
VO03	EOF/EOM ON AUDIT TRAIL DEVICE/FILE
VO04	I/O ERROR ON AUDIT TRAIL DEVICE/FILE
VO05	Reserved
VO06	I/O ERROR ON THE TAPE DURING SAVE OPERATION. TAPE HAS BEEN BACKSPACED TO THE END OF THE LAST SAVED FILE. ALL FILES ON THE IMAGE PRIOR TO THE TAPE I/O ERROR ARE SAVED ON THE TAPE.

C.48 Crash Codes

When system crash occurs as a result of a trap handler entry, the CPU halts with the registers containing the following information:

Register	Contents	
0	PSD Word 0 (when trap generated)	
1	PSD Word 1 (when trap generated)	
2	Real address of instruction causing trap	
3	Instruction causing trap	
4	CPU status word (from trap handler)	
5	Crash code:	
	MP01=X'4D503031'	(See H.IP02 Codes)
	NM01=X'4E4D3031'	(Nonpresent Memory - H.IP03)
	UI01=X'55493031'	(Undefined Instruction - H.IP04)
	PV01=X'50563031'	(Privilege Violation - H.IP05)
	MC01=X'4D433031'	(Machine Check - H.IP07)
	SC01=X'53433031'	(System Check - H.IP08)
	MF01=X'4D463031'	(Map Fault - H.IP09)
	CP01=X'43503031'	(Cache Parity Error - H.IP10) 32/67, 32/87 and 32/97
	SW01=X'53573031'	(See SWAPR codes)
6	Real address of register save block	
7	C'TRAP'=X'54524150'	

For further description, see Volume I, Chapter 2.

D Numerical Information

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5

87D13C01

2^n	n	2^{-n}
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 949 953 421 312	49	0.000 000 000 000 001 776 356 839 400 250 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0.000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0.000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0.000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0.000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0.000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0.000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5
72 057 594 037 927 936	56	0.000 000 000 000 000 013 877 787 807 814 456 755 295 395 851 135 253 906 25
144 115 188 075 855 872	57	0.000 000 000 000 000 006 938 893 903 907 228 377 647 697 925 567 626 953 125
288 230 376 151 711 744	58	0.000 000 000 000 000 003 469 446 951 953 614 188 823 848 962 783 813 476 562 5
576 460 752 303 423 488	59	0.000 000 000 000 000 001 734 723 475 976 807 094 411 924 481 391 906 738 281 25
1 152 921 504 606 846 976	60	0.000 000 000 000 000 000 867 361 737 988 403 547 205 962 240 695 953 369 140 625
2 305 843 009 213 693 952	61	0.000 000 000 000 000 000 433 680 868 994 201 773 602 981 120 347 976 684 570 312 5
4 611 686 018 427 387 904	62	0.000 000 000 000 000 000 216 840 434 497 100 886 801 490 560 173 988 342 285 156 25
9 223 372 036 854 775 808	63	0.000 000 000 000 000 000 108 420 217 248 550 443 400 745 380 086 994 171 142 578 125

87D13D01

E Powers of Integers

E.1 Powers of Sixteen in Decimal

16^n	n	16^{-n}
1	0	0.10000 00000 00000 00000 x 10
16	1	0.62500 00000 00000 00000 x 10^{-1}
256	2	0.39062 50000 00000 00000 x 10^{-2}
4 096	3	0.24414 06250 00000 00000 x 10^{-3}
65 536	4	0.15258 78906 25000 00000 x 10^{-4}
1 048 576	5	0.95367 43164 06250 00000 x 10^{-6}
16 777 216	6	0.59604 64477 53906 25000 x 10^{-7}
268 435 456	7	0.37252 90298 46191 40625 x 10^{-8}
4 294 967 296	8	0.23283 06436 53869 62891 x 10^{-9}
68 719 476 736	9	0.14551 91522 83668 51807 x 10^{-10}
1 099 511 627 776	10	0.90949 47017 72928 23792 x 10^{-11}
17 592 186 044 416	11	0.56843 41886 08080 14870 x 10^{-13}
281 474 976 710 656	12	0.35527 13678 80050 09294 x 10^{-14}
4 503 599 627 370 496	13	0.22204 46049 25031 30808 x 10^{-15}
72 057 594 037 927 936	14	0.13877 78780 78144 56755 x 10^{-16}
1 152 921 504 606 846 976	15	0.86736 17379 88403 54721 x 10^{-18}

E.2 Powers of Ten in Hexadecimal

			10^n	n	10^{-n}					
			1	0	1.0000	0000	0000	0000		
			A	1	0.1999	9999	9999	999A		
			64	2	0.28F5	C28F	5C28	F5C3	x	16^{-1}
			3E8	3	0.4189	374B	C6A7	EF9E	x	16^{-2}
			2710	4	0.68DB	8BAC	710C	B296	x	16^{-3}
		1	86A0	5	0.A7C5	AC47	1B47	8423	x	16^{-4}
		F	4240	6	0.10C6	F7A0	B5ED	8D37	x	16^{-4}
		98	9680	7	0.1AD7	F29A	BCAF	4858	x	16^{-5}
		5F5	E100	8	0.2AF3	1DC4	6118	73BF	x	16^{-6}
		3B9A	CA00	9	0.44B8	2FA0	9B5A	52CC	x	16^{-7}
	2	540B	E400	10	0.6DF3	7F67	5EF6	EADF	x	16^{-8}
	17	4876	E800	11	0.AFEB	FF0B	CB24	AAFF	x	16^{-9}
	E8	D4A5	1000	12	0.1197	9981	2DEA	1119	x	16^{-9}
	918	4E72	A000	13	0.1C25	C268	4976	81C2	x	16^{-10}
	5AF3	107A	4000	14	0.2D09	370D	4257	3604	x	16^{-11}
3	8D7E	A4C6	8000	15	0.480E	BE7B	9D58	566D	x	16^{-12}
23	86F2	6FC1	0000	16	0.734A	CA5F	6226	F0AE	x	16^{-13}
163	4578	5D8A	0000	17	0.B877	AA3	36A4	B449	x	16^{-14}
DF0	B6B3	A764	0000	18	0.1272	5DD1	D243	ABA1	x	16^{-14}
8AC7	2304	89E8	0000	19	0.1D83	C94F	B6D2	AC35	x	16^{-15}

F ASCII Interchange Code Set

Row	Col	0	1	2	3	4	5	6	7
Bit Positions									
4	0	0	0	0	0	0	0	0	0
5	1	0	0	0	0	1	1	1	1
6	2	0	0	1	1	0	0	1	1
7	3	0	1	0	1	0	1	0	1
0000	0	NUL 12-0-9-8-1	DLE 12-11-9-8-1	SP No Punch	0 0	@ 8-4	P 11-7	` 8-1	p 12-11-7
0001	1	SOH 12-9-1	DC1 11-9-1	! 12-8-7	1 1	A 12-1	Q 11-8	a 12-0-1	q 12-11-8
0010	2	STX 12-9-2	DC2 11-9-2	" 8-7	2 2	B 12-2	R 11-9	b 12-0-2	r 12-11-9
0011	3	ETX 12-9-3	DC3 11-9-3	# 8-3	3 3	C 12-3	S 0-2	c 12-0-3	s 11-0-2
0100	4	EOT 9-7	DC4 9-8-4	$ 11-8-3	4 4	D 12-4	T 0-3	d 12-0-4	t 11-0-3
0101	5	ENQ 0-9-8-5	NAK 9-8-5	% 0-8-4	5 5	E 12-5	U 0-4	e 12-0-5	u 11-0-4
0110	6	ACK 0-9-8-6	SYN 9-2	& 12	6 6	F 12-6	V 0-5	f 12-0-6	v 11-0-5
0111	7	BEL 0-9-8-7	ETB 0-9-6	' 8-5	7 7	G 12-7	W 0-6	g 12-0-7	w 11-0-6
1000	8	BS 11-9-6	CAN 11-9-8	(12-8-5	8 8	H 12-8	X 0-7	h 12-0-8	x 11-0-7
1001	9	HT 12-9-5	EM 11-9-8-1	) 11-8-5	9 9	I 12-9	Y 0-8	i 12-0-9	y 11-0-8
1010	A	LF 0-9-5	SUB 9-8-7	* 11-8-4	: 8-2	J 11-1	Z 0-9	j 12-11-1	z 11-0-9
1011	B	VT 12-9-8-3	ESC 0-9-7	+ 12-8-6	; 11-8-6	K 11-2	[12-8-2	k 12-11-2	{ 12-0
1100	C	FF 12-9-8-4	FS 11-9-8-4	0-8-3	< 12-8-4	L 11-3	\ 0-8-2	l 12-11-3	¦ 12-11
1101	D	CR 12-9-8-5	GS 11-9-8-5	- 11	= 8-6	M 11-4	] 11-8-5	m 12-11-4	} 11-0
1110	E	SO 12-9-8-6	RS 11-9-8-6	. 12-8-3	> 0-8-6	N 11-5	^ 11-8-7	n 12-11-5	~ 11-0-1
1111	F	SI 12-9-8-7	US 11-9-8-7	/ 0-1	? 0-8-7	O 11-6	_ 0-8-5	o 12-11-6	DEL 12-9-7

87D13B01

ASCII Interchange Code Set

Some positions in the ASCII code chart may have different graphic representation on various devices as:

<table>
<tr><th>ASCII</th><th>IBM 029</th></tr>
<tr><td>!</td><td>|</td></tr>
<tr><td>[</td><td>¢</td></tr>
<tr><td>]</td><td>!</td></tr>
<tr><td>^</td><td>></td></tr>
</table>

Control Characters:

NUL	-	Null
SOH	-	Start of Heading (CC)
STX	-	Start of Text (CC)
ETX	-	End of Text (CC)
EOT	-	End of Transmission (CC)
ENQ	-	Enquiry (CC)
ACK	-	Acknowledge (CC)
BEL	-	Bell (audible or attention signal)
BS	-	Backspace (FE)
HT	-	Horizontal Tabulation (punch card skip) (FE)
LF	-	Line Feed (FE)
VT	-	Vertical Tabulation (FE)
FF	-	Form Feed (FE)
CR	-	Carriage Return (FE)
SO	-	Shift Out
SI	-	Shift In
DLE	-	Data Link Escape (CC)
DC1	-	Device Control 1
DC2	-	Device Control 2
DC3	-	Device Control 3
DC4	-	Device Control 4 (stop)
NAK	-	Negative Acknowledge (CC)
SYN	-	Synchronous Idle (CC)
ETB	-	End of Transmission Block (CC)
CAN	-	Cancel
EM	-	End of Medium
SS	-	Start of Special Sequence
ESC	-	Escape
FS	-	File Separator (IS)
GS	-	Group Separator (IS)
RS	-	Record Separator (IS)
US	-	Unit Separator (IS)
DEL	-	Delete
SP	-	Space (normally nonprinting)
(CC)	-	Communication Control
(FE)	-	Format Effector
(IS)	-	Information Separator

87D13C02

G IOP/MFP Panel Mode Commands

AS	Clear address stop
AS=*xxxxxxxx*	Set address stop at address *xxxxxxxx*
BAS	Read base registers
BAS*n*=*xxxxxxxx*	Write base register *n* (0-7) with *xxxxxxxx*
CLE	Clear memory
CRMD=*xxxxxxxxxxxx* =*xxxxxxxxxxxx*	Load CRAM with *xxxxxxxxxxxx* Load CRAM with data and increment address
CS	Read control switches
CS=*xxxxxxxx*	Set control switches to *xxxxxxxx*
EA	Read effective address
EXEC	Execute CRAM
GPR	Read general purpose registers
GPR*n*=*xxxxxxxx*	Write general purpose register *n* (0-7) with *xxxxxxxx*
HALT	Halt
IPL	IPL from default address
IPL=*xxxx*	IPL from channel/subaddress *xxxx*
IS	Clear instruction stop
IS=*xxxxxxxx*	Set instruction stop at address *xxxxxxxx*
MA=*xxxxxx* <ret>	Read physical memory address location *xxxxxx* Increment and read memory address
MAV=*xxxxxx* <ret>	Read virtual memory address location *xxxxxx* Increment and read memory address
MD=*xxxxxxxx* =*xxxxxxxx* <ret>	Write memory data *xxxxxxxx* into last location addressed Increment and write memory data *xxxxxxxx* Increment and write previous data
MSGE	Message between primary and secondary panels (IOP only)
OVR	Toggle clock override

PC=*xxxxxx*	Load program counter with address *xxxxxx*
PRIP	Set primary panel (master; IOP only)
PSD	Read program status doubleword (1 and 2)
PSD=*xxxxxxxx*	Write program status word (2) with *xxxxxxxx*
PSW	Read program status word (1)
PSW=*xxxxxxxx*	Write program status word (1) with *xxxxxxxx*
RS	Clear read operand stop
RS=*xxxxxxxx*	Set read operand stop at address *xxxxxxxx*
RST	Reset
RUN	Run
SECP	Set secondary panel (master and slave; IOP only)
STEP <ret>	Instruction step Continuation of instruction step
WS	Clear write operand stop
WS=*xxxxxxxx*	Set write operand stop at address *xxxxxxxx*
@@C	Enter console mode
@@P	Enter panel mode
(LF)	Repeat command

Notes:

1. Press the return key (<ret>) after each command.
2. LOCK ON and LOCK OFF are not supported by the CRT panel.

Console Mode

To change from panel mode to console mode, enter @@C<ret>.

Upon receipt of the <ret> following the @@C command, the firmware moves the cursor on the CRT to the extreme left margin of the next line.

To return to the panel mode, enter @@P<ret>. When the panel mode is selected, // is the prompt.

H Standard Date and Time Formats

H.1 Description

With the advent of the new MPX-32 file system, proper maintenance of the system date and time becomes more important than ever before as all file system resources will be time stamped to aid in management. It is vital the date and time be kept in a manner that is at once useful in this application and also convenient to convert into other formats that the user might require.

System date and time are kept in standard binary format. This format consists of two words: the first word contains the date and the second word contains the time. The date is maintained as the number of days since January 1, 1960 and the time is maintained as the binary count of system time units since midnight, adjusted to 100 microsecond granularity.

For the convenience of the user, monitor service calls are provided to convert the date and time between any of three standard formats. These are:

1. Binary Format (described above)
2. Byte Binary Format
3. ASCII Format (sometimes referred to as quad ASCII format)

Byte binary format time consists of two words: the first word contains date information and the second word contains time information. In byte binary format, the date is kept as four distinct values instead of one. Byte 0 of the date word is the binary century, byte 1 is the binary year in that century, byte 2 is the binary month and byte 3 the binary day of the month. Time is kept in a similar manner with byte 0 being the hour, byte 1 the minute, byte 2 the second, and byte 3 the number of clock ticks.

ASCII format consists of four words of information. The first two words contain the ASCII century, year, month, and day in successive halfwords. The second two words contain the hour, minutes, seconds, and clock ticks in a similar fashion. In ASCII format, use of a 120-hertz clock can cause truncation of the clock tick fields, allowing for only two ASCII digits.

H.2 Date/Time Standard Formats

Binary

Date	Time
Days since 1/1/60	Clock ticks since midnight
Word 1	Word 2

Byte Binary

Date				Time			
Bin Cent.	Bin Year	Bin Month	Bin Day	Bin Hour	Bin Min.	Bin Sec.	Bin Ints.
Word 1				Word 2			

Quad ASCII

Date

Century	Year	Month	Day
Word 1		Word 2	

Time

Hour	Minute	Second	Interrupt
Word 3		Word 4	

I Compressed Source Format

Compressed source files are blocked files that consist of 120 byte records. The last record may be less than 120 bytes and has a data type code of 9F. The structure of a compressed record is described below.

Each record contains 6 control bytes:

1 byte	data type code, BF (9F indicates last record)
1 byte	byte count, number of data bytes in record
2 bytes	checksum, halfword sum of data bytes
2 bytes	sequence number, record sequence number starting at zero

Data is recorded as follows:

1 byte	blank count, number of blanks before data
1 byte	data count, number of data bytes
n-bytes	actual ASCII data

.
. (this sequence is repeated until the end of a line is reached)
.

1 byte	EOL character, FF

J Map Block Address Assignments

Map Block # Decimal/Hex	Page # Decimal/Hex	Address Range Hexadecimal
00/00	00/00	00000 - 01FFF
01/01	04/04	02000 - 03FFF
02/02	08/08	04000 - 05FFF
03/03	12/0C	06000 - 07FFF
04/04	16/10	08000 - 09FFF
05/05	20/14	0A000 - 0BFFF
06/06	24/18	0C000 - 0DFFF
07/07	28/1C	0E000 - 0FFFF
08/08	32/20	10000 - 11FFF
09/09	36/24	12000 - 13FFF
10/0A	40/28	14000 - 15FFF
11/0B	44/2C	16000 - 17FFF
12/0C	48/30	18000 - 19FFF
13/0D	52/34	1A000 - 1BFFF
14/0E	56/38	1C000 - 1DFFF
15/0F	60/3C	1E000 - 1FFFF
16/10	64/40	20000 - 21FFF
17/11	68/44	22000 - 23FFF
18/12	72/48	24000 - 25FFF
19/13	76/4C	26000 - 27FFF
20/14	80/50	28000 - 29FFF
21/15	84/54	2A000 - 2BFFF
22/16	88/58	2C000 - 2DFFF
23/17	92/5C	2E000 - 2FFFF
24/18	96/60	30000 - 31FFF
25/19	100/64	32000 - 33FFF
26/1A	104/68	34000 - 35FFF
27/1B	108/6C	36000 - 37FFF
28/1C	112/70	38000 - 39FFF
29/1D	116/74	3A000 - 3BFFF
30/1E	120/78	3C000 - 3DFFF
31/1F	124/7C	3E000 - 3FFFF
32/20	128/80	40000 - 41FFF
33/21	132/84	42000 - 43FFF
34/22	136/88	44000 - 45FFF
35/23	140/8C	46000 - 47FFF
36/24	144/90	48000 - 49FFF
37/25	148/94	4A000 - 4BFFF

Map Block # Decimal/Hex	Page # Decimal/Hex	Address Range Hexadecimal
38/26	152/98	4C000 - 4DFFF
39/27	156/9C	4E000 - 4FFFF
40/28	160/A0	50000 - 51FFF
41/29	164/A4	52000 - 53FFF
42/2A	168/A8	54000 - 55FFF
43/2B	172/AC	56000 - 57FFF
44/2C	176/B0	58000 - 59FFF
45/2D	180/B4	5A000 - 5BFFF
46/2E	184/B8	5C000 - 5DFFF
47/2F	188/BC	5E000 - 5FFFF
48/30	192/C0	60000 - 61FFF
49/31	196/C4	62000 - 63FFF
50/32	200/C8	64000 - 65FFF
51/33	204/CC	66000 - 67FFF
52/34	208/D0	68000 - 69FFF
53/35	212/D4	6A000 - 6BFFF
54/36	216/D8	6C000 - 6DFFF
55/37	220/DC	6E000 - 6FFFF
56/38	224/E0	70000 - 71FFF
57/39	228/E4	72000 - 73FFF
58/3A	232/E8	74000 - 75FFF
59/3B	236/EC	76000 - 77FFF
60/3C	240/F0	78000 - 79FFF
61/3D	244/F4	7A000 - 7BFFF
62/3E	248/F8	7C000 - 7DFFF
63/3F	252/FC	7E000 - 7FFFF
Extended Memory 128KW to 256KW - 1B		
64/40	256/100	80000 - FFFFF
Extended Memory 256KW to 384KW - 1B		
128/80	512/200	100000 - 17FFFF
Extended Memory 384KW to 512KW - 1B		
192/C0	768/300	180000 - 1FFFFF
Extended Memory 512KW to 1024KW - 1B		
256/100	1024/400	200000 - 3FFFFF
Extended Memory 1024KW to 2048KW - 1B		
512/200	2048/800	400000 - 7FFFFF
Extended Memory 2048KW to 4096KW - 1B		
1024/400	4096/1000	800000 - FFFFFF

K Control Switches

While rebooting the system, various initialization processes can be inhibited or enabled by setting the appropriate control switches. The switch assignments are:

Switch	Function if Set
0	Inhibits volume clean-up by J.MOUNT.
1	SYSINIT enters the system debugger before processing patches.
2	Inhibits patch processing (see Reference Manual, Volume III, Chapter 9, Entry Conditions).
3	Inhibits terminal initialization.
4	Inhibits accounting functions including the M.KEY, M.PRJCT, M.ACCNT, and M.ERR files.
5	Inhibits processing of the sequential task activation table at IPL time.
6	If J.MOUNT encounters an invalid resource descriptor due to an invalid resource descriptor type field or space definition, it branches and links to the system debugger (if present) with R2 pointing to the resource descriptor.
7	J.MOUNT prereads the file space bit map (SMAP) or the resource descriptor allocation bit map (DMAP). J.MOUNT will not perform file overlap protection.
8	Delete spooled output files instead of resubmitting them for processing.
9	Inhibits activating LOADACS during IPL or RESTART operations.
10	Enables faster memory initialization by checking only one location per map block to determine if that map block is present. It is not recommended that this switch be set on the first IPL after power up.
11	Inhibits initialization of the memory descriptor table (MDT).
12	For RMSS: inhibits booting of nodes while J.BOOT executes.

The control switches can be accessed by the console. The proper time to set the switches is while the system is waiting for the date and time to be entered. To set, for example, switch 3, the following must be entered on the IOP/MFP console:

```
ENTER DATE AND TIME: @@P
//CS=10000000    Terminal Initialization Inhibited
//@@C
<CR>
INVALID DATE FORMAT=MM/DD/XX
ENTER DATE AND TIME:
```

Refer to the CONCEPT 32/2000 Operations manual for instructions for setting control switches on the Amiga console.

During power up, control switches are prezeroed if the proper firmware revision level has been installed. Power up without prezeroing can cause unexpected system responses due to incorrect control settings.

All control switch settings are preserved during system reboots not involving system power up (i.e., online restart and IPL).

L Data Structures

L.1 Introduction

This appendix contains some of the more frequently used data structures. Below is a list of those structures.

Caller Notification Packet (CNP)
Controller Definition Table (CDT)
Dispatch Queue Entry (DQE)
File Control Block (FCB), 16 Word
File Control Block (FCB), 8 Word
File Control Block (FCB), High Speed Data
File Pointer Table (FPT)
Parameter Task Activation Block (PTASK)
TSM Procedure Call Block (PCB)
Pathname Blocks (PNB)
Post Program-Controlled Interrupt Notification Packet (PPCI)
Parameter Receive Block (PRB)
Parameter Send Block (PSB)
Resource Create Block (RCB)
Resource Identifiers (RID)
Resource Logging Block (RLB)
Resource Requirement Summary (RRS) Entries
Receiver Exit Block (RXB)
Type Control Parameter Block (TCPB)
Unit Definition Table (UDT)

L.2 Caller Notification Packet (CNP)

The caller notification packet (CNP) is the mechanism used by the Resource Management Module (H.REMM) and the Volume Management Module (H.VOMM) for handling abnormal conditions that may result during resource requests. All or part of this structure can be used by a particular service being called. The CNP must be on a word boundary.

	0 — 7	8 — 15	16 — 23	24 — 31
Word 0	Time-out value (CP.TIMO)			
1	Abnormal return address (CP.ABRET)			
2	Option field (CP.OPTS). See Note 1.		Status field (CP.STAT). See Note 2.	
3-4	Reserved (See Note 3.)			
5	Automatic open FCB address (CP.FCBA)			

Notes:

1. A bit sequence and/or value used to provide additional information that can be necessary to fully define the calling sequence for a particular service.
2. A right-justified numeric value identifying the return status for this call.
3. Refer to the individual system service description in the MPX-32 Reference Manual Volume I for interpretation of these words.

L.3 Controller Definition Table (CDT)

The controller definition table (CDT) is a system resident structure used to identify information required by handlers and the I/O processor for a specific controller. The CDT is built by the SYSGEN process, one for each controller configured on the system. The CDT identifies devices (UDTs) associated with the controller, the handler address associated with the controller, and defines other pertinent controller information.

<table>
<tr><th></th><th>0 7</th><th>8 15</th><th>16 23</th><th>24 31</th></tr>
<tr><td>Word 0</td><td colspan="4">String forward address (CDT.FIOQ)</td></tr>
<tr><td>1</td><td colspan="4">String backward address (CDT.BIOQ)</td></tr>
<tr><td>2</td><td>Link priority (CDT.LPRI). See Note 1.</td><td>Number of entries in list (CDT.IOCT). See Note 2.</td><td>Class (CDT.CLAS). See Note 3.</td><td>Flags (CDT.FLG2). See Note 4.</td></tr>
<tr><td>3</td><td colspan="2">CDT index (CDT.INDX)</td><td>Device type code (CDT.DTC) See Note 5.</td><td>Interrupt priority level (CDT.IPL)</td></tr>
<tr><td>4</td><td>Number units on controller (CDT.NUOC)</td><td>Number requests outstanding (CDT.IORO)</td><td>Channel number (CDT.CHAN)</td><td>Subaddress of first device (CDT.SUBA)</td></tr>
<tr><td>5</td><td>Program number if reserved (CDT.PNRC)</td><td colspan="3">Interrupt handler address (CDT.SIHA) or controller information block (CDT.CIF)</td></tr>
<tr><td>6</td><td>Flags (CDT.FLGS). See Note 6.</td><td colspan="3">UDT address of first device on controller (CDT.UDTA)</td></tr>
<tr><td>7</td><td>I/O status (CDT.IOST). See Note 7.</td><td colspan="3">TI address (CDT.TIAD)
or
SI address if extended I/O (CDT.SIAD)</td></tr>
<tr><td>8</td><td colspan="4">UDT address unit 0* (CDT.UT0)</td></tr>
<tr><td>9-23</td><td colspan="4">UDT address unit 1* (CDT.UT1) through
UDT address unit 15* (CDT.UTF)</td></tr>
</table>

*Initialized by SYSGEN

Notes:

1. Always zero (head cell)
2. Number of entries in list (zero if none)

3. Values in CDT.CLAS are assigned as follows:

Value	Meaning
X'0D'	TCW type with extended addressing capability
X'0E'	TCW type
X'0F'	extended I/O

4. Bits in CDT.FLG2 are assigned as follows:

Bit	Meaning if Set
0	SCSI device (CDT.SCSI)
1-7	reserved for future use

5. For example, 01 for any disk, 04 for any tape, etc. Valid device type codes are listed in Appendix A.

6. Bits in CDT.FLGS are assigned as follows:

Bit	Meaning if Set
0	extended I/O device (CDT.FCLS)
1	I/O outstanding (set by handler, reset by IOCS) (CDT.IOU1)
2	GPMC device (CDT.GPMC)
3	initialization (INC) needs to be performed for this controller (CDT.FINT)
4	D-class (CDT.XGPM)
5	used only when IOQs are linked to the CDT. Set when SIO is accepted by the controller. Reset when IOQ is unlinked from the CDT or when I/O is reported complete to IOCS in the case of operator intervention type errors (CDT.IOU5).
6	IOP controller (CDT.IOP)
7	controller malfunction (CDT.MALF)

7. Bits in CDT.IOST are assigned as follows:

Bit	Meaning if Set
0	IOQ linked to UDT (CDT.NIOQ)
1	multiplexing controller (CDT.MUXC)
2	use standard XIO interface
3	16MB GPMC (CDT.XGPS)
4	cache controller (CDT.CAC)
5	H.F8XIO has determined if the controller is pre-8512-2 or not (CDT.CKFL)
6	controller not pre-8512-2 (CDT.FLOW)
7	reserved for FMS

L.4 Dispatch Queue Entry (DQE)

The dispatch queue entry (DQE) contains all of the core-resident information required to describe an active task to the system. It is always linked to the CPU scheduler state chain that describes the current execution status of the associated task.

Word No. (Decimal)	Byte (Hex)	0 – 7	8 – 15	16 – 23	24 – 31
0	0	DQE.SF			
1	4	DQE.SB			
2	8	DQE.CUP	DQE.BUP	DQE.IOP	DQE.US
3	C	DQE.NUM/DQE.TAN			
4-5	10	DQE.ON			
6-7	18	DQE.LMN			
8-9	20	DQE.PSN			
10	28	DQE.USW			
11	2C	DQE.USHF			
12	30	DQE.MSD			
13	34	DQE.KCTR			
14	38	DQE.MMSG	DQE.MRUN	DQE.MNWI	DQE.GQFN
15	3C	DQE.UF2	DQE.IPUF	DQE.NWIO	DQE.SOPO
16	40	DQE.CQC			
17	44	DQE.SH	DQE.SHF	DQE.TIFC	DQE.RILT
18	48	DQE.UTS1			
19	4C	DQE.UTS2			
20	50	DQE.DSW			
21	54	DQE.PRS			
22	58	DQE.PRM			
23	5C	Reserved	DQE.TSKF	DQE.MSPN	DQE.MST
24	60	DQE.PSSF			
25	64	DQE.PSSB			
26	68	DQE.PSPR	DQE.PSCT	DQE.ILN	DQE.RESU
27	6C	DQE.TISF			
28	70	DQE.TISB			
29	74	DQE.TIPR	DQE.TICT	DQE.SWIF	DQE.UBIO
30	78	DQE.RRSF			
31	7C	DQE.RRSB			
32	80	DQE.RRPR	DQE.RRCT	DQE.NSCT	
33	84	DQE.MRSF			
34	88	DQE.MRSB			

Dispatch Queue Entry (DQE)

Word No. (Decimal)	Byte (Hex)	0 – 7	8 – 15	16 – 23	24 – 31
35	8C	DQE.MRPR	DQE.MRCT	DQE.NWRR	DQE.NWMR
36	90	DQE.RTI	DQE.NWLM	DQE.ATI	Reserved
37	94	DQE.SAIR/DQE.TAD			
38-40	98	DQE.ABC			
41	A4	DQE.TSAP			
42-43	A8	DQE.SRID/DQE.PGOL			
	AC	DQE.SRID/DQE.PGOC		DQE.SRID/Reserved	
44-51	B0	DQE.CDIR/DQE.CVOL			
52	D0	DQE.GID	Reserved	DQE.ASH	
53	D4	DQE.ACX2			
54	D8	DQE.MRQ	DQE.MEM	DQE.MEMR	
55	DC	DQE.MRT	Reserved	DQE.RMMR	
56	E0	DQE.MAPN		DQE.CME	
57	E4	DQE.CMH		DQE.CMS	
58-63	FC	Reserved			

Byte (Hex)	Symbol	Description
0	DQE.SF	String forward linkage address; Field length = 1W; Standard linked list format; Contains address of next (top-to-bottom) entry in chain.
4	DQE.SB	String backward linkage address; Field length = 1W; Standard linked list format; Contains address of next (bottom-to-top) entry in chain.
8	DQE.CUP	Current user priority; Standard linked list format; This priority is adjusted for priority migration based on situational priority increments. Situational priority increments are based on the base level priority (DQE.BUP) of the task.
	DQE.BUP	Base priority of user task; Field length = 1B; Used by scheduler to generate DQE.CUP (current priority) based on any situational priority increments.
	DQE.IOP	I/O priority; Field length = 1B; Initially set from base priority; Used for I/O queue priority.

Byte (Hex)	Symbol	Description
	DQE.US	State chain index for this user task; Field length = 1B; Range: zero through X'1E'; Indicates current state of this task, such as ready-to-run priority, I/O wait, resource block, etc.

Label	Index	Task description
FREE	00	DQE is available (in free list)
PREA	01	activation in progress
CURR	02	currently executing task or is pre-empted time-distribution task in quantum stage one
SQRT	03	ready to run (priority level 1 to 54)
SQ55	04	ready to run (priority level 55)
SQ56	05	ready to run (priority level 56)
SQ57	06	ready to run (priority level 57)
SQ58	07	ready to run (priority level 58)
SQ59	08	ready to run (priority level 59)
SQ60	09	ready to run (priority level 60)
SQ61	0A	ready to run (priority level 61)
SQ62	0B	ready to run (priority level 62)
SQ63	0C	ready to run (priority level 63)
SQ64	0D	ready to run (priority level 64)
SWTI	0E	waiting for terminal input
SWIO	0F	waiting for I/O
SWSM	10	waiting for message complete
SWSR	11	waiting for run request complete
SWLO	12	waiting for low speed output
SUSP	13	waiting for timer expiration, resume request, or message interrupt
RUNW	14	waiting for timer expiration, or run request
HOLD	15	waiting for a continue request
ANYW	16	waiting for timer expiration, no-wait I/O complete, no-wait message complete, no-wait run request complete, message interrupt, or break interrupt
SWDC	17	waiting for disk space
SWDV	18	waiting for device allocation
SWFI	19	waiting for file system
MRQ	1A	waiting for memory
SWMP	1B	waiting for memory pool
SWGQ	1C	waiting in general wait queue
CIPU	1D	current IPU task in execution
RIPU	1E	IPU requesting state

Dispatch Queue Entry (DQE)

Byte (Hex)	Symbol	Description
C	DQE.NUM	DQE entry number; Field length = 1B; Used as an index to DQE address table (DAT); Range: one through "N"(for MPL index compatibility); Used by scheduler to set C.PRNO to reflect the currently executing task. This value is also used as the MPL index. It is used by the scheduler to initialize the CPIX in the PSD before loading the map for this task.
	DQE.TAN	Task activation sequence number; Field length = 1W; This number is assigned by the activation service and uniquely identifies a task. **Note:** The most significant byte of this value is the DQE entry number and is accessible as DQE.NUM.
10	DQE.ON	Owner name; Field length = 1D.
18	DQE.LMN	Load module name; Field length = 1D.
20	DQE.PSN	Pseudonym associated with task; Field length = 1D; This parameter is an optional argument accepted by the pseudo task activation service. It can be used to uniquely identify a task within a subsystem, such as multibatch. It contains descriptive information useful to the system operator or to other tasks within a subsystem. Conventions used to generate a pseudonym are determined by the associated subsystem. A system-wide convention should be used to establish pseudonym prefix conventions to avoid confusion between subsystems.
28	DQE.USW	User status word; Field length = 1W.
2C	DQE.USHF	Scheduling flags; Field length = 1W; Used by the scheduler to indicate special status conditions.

Byte (Hex)	Symbol	Description

Bit	Meaning When Set
00	load protection image requested (DQE.LPI)
01	single copy load module (DQE.SING)
02	task is indirectly connected (DQE.INDC)
03	task is privileged (DQE.PRIV)
04	task has message receiver (DQE.MSGR)
05	task has break receiver (DQE.BRKR)
06	task quantum stage one expired (DQE.QS1X)
07	task quantum stage two expired (DQE.QS2X)
08	in-swap I/O error (DQE.INER)
09	wait I/O request outstanding (DQE.WIOA)
10	wait I/O complete before in-progress notification (DQE.WIOC)
11	inhibit message pseudointerrupt (DQE.INMI)
12	batch origin task (DQE.BAOR)
13	running in TSM environment (DQE.TMOR)
14	task abort in progress (DQE.ABRT)
15	task is in pre-exit state (DQE.PRXT)
16	run receiver mode (DQE.RRMD)
17	wait send message outstanding (DQE.WMSA)
18	wait message complete before link to wait queue (DQE.WMSC)
19	wait mode send run request outstanding (DQE.WRRA)
20	wait mode send run request complete before link to wait queue (DQE.WRRC)
21	debug associated with task (DQE.DBAT)
22	real-time task (DQE.RT)
23	time-distribution task initial dispatch (DQE.TDID) Set by: • H.ALOC1 on activation of T/D task. • S.EXEC51 when task is linked to wait state. • H.EXEC7 on completion of inswap or other memory request. Reset by: • S.EXEC20 on initial dispatch of task after activation • Wait state termination • In-swap
24	task delete in progress (DQE.DELP)
25	task abort (with abort receiver) in progress (DQE.ABRA)
26	abort receiver established (DQE.ABRC)
27	asynchronous abort/delete inhibited (DQE.ADIN)
28	asynchronous delete deferred (DQE.ADDF)
29	task is inactive (DQE.INAC)
30	asynchronous abort deferred (DQE.AADF)
31	activation timer in effect (DQE.ACTT)

Byte (Hex)	Symbol	Description
30	DQE.MSD	Physical address of MIDL in TSA; Field length = 1W.
34	DQE.KCTR	Kill/abort timer; Field length = 1W.
38	DQE.MMSG	Maximum number of no wait messages allowed to be sent by this task; Field length = 1B.
	DQE.MRUN	Maximum number of no-wait run requests allowed to be sent by this task; Field length = 1B.
	DQE.MNWI	Maximum number of no-wait I/O requests allowed to be concurrently outstanding for this task; Field length = 1B.
	DQE.GQFN	Contains the generalized queue (SWGQ) function code; Field length = 1B; Function codes are queued as follows:

Code	Meaning
01	volume resource (QVRES)
02	ART space (QART)
03	mount in progress (QMNT)
04	resourcemark lock (QRSM)
05	reserved for eventmark (QEVM)
06	read wait for writer (QGEN)
07	shared memory table (QSMT)
08	synchronous resource lock (QSRL)
09	mounted volume table (QMVT)
0A	dual-port lock (QDPLK)
0B	suspend dual-port lock (QSUSP)
0C	debug wait (QDBGW)
0D	remote message area (QMSG)
0E	remote message event (QSER)
0F	remote allocate area (QASMP)
10	remote deallocate area (QDSMP)
11	remote abort area (QAMSG)
12	remote enable/disable area (QOMSG)
13	wait for TSM (QWTSM)

Byte (Hex)	Symbol	Description
3C	DQE.UF2	Scheduling flags; Field length = 1B;

Bit	Meaning if Set
0	enable debug mode break (DQE.EDB)
1	generalized wait queue time-out (DQE.GQTO)
2	task interrupts are synchronized (DQE.SYNC)
3	task is part of a job (DQE.JOB)
4	ACX-32 task flag (DQE.ACX)
5	special arithmetic function requested (DQE.AF)
6	reserved
7	run request terminated (DQE.RRT)

Byte (Hex)	Symbol	Description
	DQE.IPUF	IPU flag byte; Field length = 1B;

Bit	Meaning if Set
0	IPU inhibit flag (DQE.IPUH)
1	IPU bias flag (DQE.IPUB)
2	CPU only (DQE.IPUR)
3	OS execution direction flag (set when PSD is in user area) (DQE.OSD)
4	base register task (DQE.BASE)
5	Ada task (DQE.ADA)
6	PTRACE debugger task (DQE.PDBG)
7	H.PTRAC task association control bit (DQE.PTRA)

Byte (Hex)	Symbol	Description
	DQE.NWIO	Number of no-wait I/O requests; Field length = 1B.
	DQE.SOPO	Priority bias only swapping control flags; Field length = 1B;

Bit	Meaning if Set
0	SWGQ state priority-based swapping (DQE.GQPO)
1	swap inhibit due to bit map access (DQE.BMAP)
2	inhibit swap device while accessing MDT (DQE.MDTA)
3	user swap inhibit flag (DQE.USWI)
4	user swap on priority only flag (DQE.USPO)
5-7	reserved

Dispatch Queue Entry (DQE)

Byte (Hex)	Symbol	Description
40	DQE.CQC	Current quantum count; Field length = 1W; Used by the scheduler to accumulate elapsed execution time for the task to compare the level unique stage one and stage two time-distribution values.
44	DQE.SH	Used by J.SWAPR to swap shadow memory; Field length = 1B.
	DQE.SHF	Shadow memory flag; Field length = 1B;

Bit	Meaning if Set
0	task requests shadow memory (DQE.SHAD)
1	IPU shadow memory requested (DQE.SHI)
2	IPU/CPU Common Shadow Memory requested (DQE.SHB).

Byte (Hex)	Symbol	Description
	DQE.TIFC	Timer function code; Field length = 1B;

Value	Meaning
00	not active
01	request interrupt
02	resume program from suspend (SUSP) queue
03	resume program from any-wait (ANYW) queue
04	resume program from run-request-wait (RUNW) queue
05	resume program from generalized (SWGQ) queue
06	resume program from peripheral device (SWDV) queue
07	resume program from disk space (SWDC) queue

Byte (Hex)	Symbol	Description
	DQE.RILT	Request Interrupt (RI) level for timer; Field length = 1B; Identifies the interrupt level to be requested upon timer expiration.
48	DQE.UTS1	User timer slot word 1; Field length = 1W; Current timer value; Contains negative number of timer units before time out.

Byte (Hex)	Symbol	Description
4C	DQE.UTS2	User timer slot word 2; Field length = 1W; Reset timer value; Contains negative number of timer units; Used to reset the current timer value when it expires.
50	DQE.DSW	Base mode debugger status word (PCALL); Field length = 1W.
54	DQE.PRS	Peripheral requirement specification; Field length = 1W;

Bit	Description
0-7	reserved
8-15	device type code
16-23	channel address
24-31	subchannel address or contains first word of SWGQ ID.

Byte (Hex)	Symbol	Description
58	DQE.PRM	Peripheral requirements mask; Field length = 1W;

Value	Meaning
X'00FF0000'	any device of this type code
X'00FFFF00'	any device of the specified type code on the specified channel
X'00FFFFFF'	the specified device as described by type code, channel, and subchannel address, or contains second word of SWGQ ID.

Byte (Hex)	Symbol	Description
5C	Reserved	Field length = 1B
	DQE.TSKF	Task flags; Field length = 1B;

Bit	Meaning if Set
0	real-time accounting disabled (DQE.RTAC)
1-2	reserved for RMSS
3	task is running with MPX-32 mapped out (DQE.MAPO)
4	reserved for MPX-32
5	task is demand paged (DQE.DPG)
6	inhibit page out (DQE.NPGO)
7	reserved

Dispatch Queue Entry (DQE)

Byte (Hex)	Symbol	Description
	DQE.MSPN	TSA maps required to span MIDLs and MEMLs; Field length = 1B.
	DQE.MST	Static memory type specification; Field length = 1B; Value — Memory Class 01 — E 02 — H 03 — S 04 — H1 05 — H2 06 — H3 This field is used to specify the type of memory required for in-swap.
60	DQE.PSSF	Pre-emptive system service head cell string forward linkage address; Standard head cell format; Field length = 1W; Contains address of next (top-to-bottom) entry in chain.
64	DQE.PSSB	Pre-emptive system service head cell string backward linkage address; Standard head cell format; Field length = 1W; Contains address of next (bottom-to-top) entry in chain.
68	DQE.PSPR	Pre-emptive system service head cell dummy priority (always zero); Standard head cell format; Field length = 1B.
	DQE.PSCT	Pre-emptive system service head cell number of entries in list; Standard head cell format; Field length = 1B.
	DQE.ILN	Interrupt level number; Field length = 1B; Identifies associated interrupt level for interrupt connected tasks.
	DQE.RESU	Reserved usage index; Field length = 1B.

Byte (Hex)	Symbol	Description
6C	DQE.TISF	Task interrupt head cell string forward linkage address; Standard head cell format; Field length = 1W; Contains address of next (top-to-bottom) entry in chain.
70	DQE.TISB	Task interrupt head cell string backward linkage address; Standard head cell format; Field length = 1W; Contains address of next (bottom-to-top) entry in chain.
74	DQE.TIPR	Task interrupt head cell dummy priority (always zero); Standard head cell format; Field length = 1B.
	DQE.TICT	Task interrupt head cell number of entries in list; Standard head cell format; Field length = 1B.
	DQE.SWIF	Swapping inhibit flags; Field length = 1B;

Bit	Task Meaning if Set
0	resident (DQE.RESP)
1	locked in memory (DQE.LKIM)
2	unbuffered I/O in progress (DQE.IO)
3	outswapped (DQE.OTSW)
4	leaving system (DQE.TLVS)
5	forced unswappable during terminal output (DQE.FCUS)
6	forced unswappable because swap file has not been allocated for it (DQE.FCRS)
7	imbedded in the operating system (DQE.INOS)

Byte (Hex)	Symbol	Description
	DQE.UBIO	Number of unbuffered I/O requests currently outstanding; Field length = 1B.
78	DQE.RRSF	Run receiver head cell string forward linkage address; Standard head cell format; Field length = 1W; Contains address of next (top-to-bottom) entry in chain.
7C	DQE.RRSB	Run receiver head cell string backward linkage address; Standard head cell format; Field length = 1W; Contains address of next (bottom-to-top) entry in chain.
80	DQE.RRPR	Run receiver head cell dummy priority (always zero); Standard head cell format; Field length = 1B.

Byte (Hex)	Symbol	Description
	DQE.RRCT	Run receiver head cell number of entries in list; Standard head cell format; Field length = 1B.
	DQE.NSCT	Number of map blocks outswapped; Field length = 1H.
84	DQE.MRSF	Message receiver head cell string forward Linkage address; Standard head cell format; Field length = 1W; Contains address of next (top-to-bottom) entry in chain.
88	DQE.MRSB	Message receiver head cell string backward Linkage address; Standard head cell format; Field length = 1W; Contains address of next (bottom-to-top) entry in chain.
8C	DQE.MRPR	Message receiver head cell dummy priority (always zero); Standard head cell format; Field length = 1B.
	DQE.MRCT	Message receiver head cell number of entries in list; Standard head cell format; Field length = 1B.
	DQE.NWRR	Number of no-wait mode run requests outstanding; Field length = 1B.
	DQE.NWMR	Number of no-wait mode message requests outstanding; Field length = 1B.
90	DQE.RTI	Requested task interrupt flags; Field length = 1B;

Bit	Meaning if Set
0	reserved
1	priority one end action request. Used for pre-emptive system services. (DQE.EA1R)
2	debug break request (DQE.DBBR)
3	user break request (DQE.UBKR)
4	priority two end action request (DQE.EA2R)
5	message interrupt request (DQE.MSIR)
6-7	reserved

Byte (Hex)	Symbol	Description
	DQE.NWLM	No-wait run request limit. Field length = 1B.
	DQE.ATI	Active task interrupt flags; Field length = 1B;

Bit	Meaning if Set
0	reserved
1	priority one active end action (DQE.AEA1)
2	active debug break (DQE.ADM)
3	active user break (DQE.AUB)
4	priority two active end action (DQE.AEA)
5	active message interrupt (DQE.AMI)
6-7	reserved

Byte (Hex)	Symbol	Description
	Reserved	Field length = 1B.
94	DQE.SAIR	System action task interrupt request;

Bit	Meaning if Set
0	request for delete of this task (DQE.DELR)
1	reserved
2	hold task request (DQE.HLDR)
3	abort task request (DQE.ABTR)
4	exit task request (DQE.EXTR)
5	suspend task request (DQE.SUSR)
6	run receiver mode request (DQE.RRRQ)
7	reserved

Byte (Hex)	Symbol	Description
	DQE.TAD	TSA address (logical); Field length = 1W; Byte zero contains DQE.SAIR.
98	DQE.ABC	Abort code; Field length = 3W.
A4	DQE.TSAP	Physical address of the TSA
A8-AC	DQE.SRID	If DQE.DPG is reset; Used swap space linked list; Field length = 2W.
	DQE.PGOL	If DQE.DPG is set; Forward pointer to MPTL (MAP.SF); Field length = 1HW Backward pointer to MPTL (MAP.SB) Field length = 1HW
	DQE.PGOC	Number of pages queued for pageout Field length = 1HW
	Reserved	Field length = 1HW

Dispatch Queue Entry (DQE)

Byte (Hex)	Symbol	Description
B0	DQE.CDIR	Load module RID at activation; Field length = 8W.
	DQE.CVOL	Current working volume at activation; Field length = 8W.
D0	DQE.GID	Group swap identification; Field length = 1B.
D1	Reserved	1 Byte
D2	DQE.ASH	Number of shadow memory blocks currently allocated Field length = 1H.
D4	DQE.ACX2	Advance communication word; Field length = 1W.
D8	DQE.MRQ	Memory request doubleword; Reserved field length = 1B.
	DQE.MEM	Type of memory requested; Field length = 1B;

Value	Memory Class
01	E
02	H
03	S

Byte (Hex)	Symbol	Description
	DQE.MEMR	Number of memory blocks required; Field length = 1H.
DC	DQE.MRT	Memory request type code; Field length = 1B;

Value	Meaning
00	in-swap only
01	preactivation request
02	activation request
03	memory expansion request
04	IOCS buffer request
05	shared memory request
06	system buffer request
07	release swap file space

If DQE.MRT equals 05, the next three bytes will contain the address of the shared memory table entry.

Byte (Hex)	Symbol	Description
	Reserved	Field length = 1B.
	DQE.RMMR	Map register for requested memory; Field length = 1H.

Byte (Hex)	Symbol	Description
E0	DQE.MAPN	Inclusive span of maps in use; Field length = 1H.
	DQE.CME	Number of swappable class E map blocks currently allocated; For resident tasks, if not zero, reflects the total number of map blocks in use. Field length = 1H.
E4	DQE.CMH	Number of swappable class H map blocks currently allocated; For resident tasks, if not zero, reflects the total number of map blocks in use. Field length = 1H.
	DQE.CMS	Number of swappable class S map blocks currently allocated; For resident tasks, if not zero, reflects the total number of map blocks in use. Field length = 1H.
E8	Reserved	Reserved for MPX-32

L.5 File Control Block (FCB), 16 Word

Word	0 7	8 12	13 31
0	**Opcode (FCB.OPCD)**	Logical file code (FCB.LFC)	
1	**Reserved**		
2	General control flags (FCB.GCFG)	Special flags (FCB.SCFG)	**Reserved**
3	**Status flags (FCB.SFLG)**		
4	**Actual transfer quantity (FCB.RECL)**		
5	**Reserved**	**I/O queue address (FCB.IOQA)**	
6	**Special Status (FCB.SPST)**	Wait I/O error return address (FCB.ERRT)	
7	**Index to FPT (FCB.FPTI)**	**FAT address (FCB.FATA)**	
8	**Reserved**	I/O buffer address (FCB.ERWA)	
9	Transfer quantity (bytes) (FCB.EQTY)		
10	Random access address (FCB.ERAA)		
11	**Extended I/O status word one (FCB.IST1)**		
12	**Extended I/O status word two (FCB.IST2)**		
13	**Reserved**	No-wait I/O normal end-action service address (FCB.NWOK)	
14	**Reserved**	No-wait I/O error end-action service address (FCB.NWER)	
15	Number of buffers (FCB.BBN)	Address of blocking buffer (FCB.BBA)	

Shaded areas are set by the system.

T1FCB

Word 0

Bit 0 Reserved

Bits 1-7 Operation code (FCB.OPCD) — type of function requested of the device handler. This field is set by IOCS as a function of the executed service.

Bits 8-31 Logical file code (FCB.LFC) — any combination of three ASCII characters is allowed. The LFC must match the previously assigned LFC of the I/O resource being accessed.

Word 1

Bits 0-31 Reserved

Word 2

Bits 0-7 General control flags (FCB.GCFG) — these eight bits enable the user to specify the manner in which an operation is to be performed by IOCS. The interpretation of these bits is shown as follows:

Bit	Meaning if Set	Definition
0	NWT	IOCS returns to the user immediately after the I/O operation is queued. If reset, IOCS exits to the calling program only when the requested operation has been completed.
1	NER	error processing is not performed by either the device handler or IOCS. An error return address is ignored and a normal return is taken to the caller; however, the device status is posted in the FCB unless bit 3 is set. If reset, normal error recovery is attempted. Normal error processing for disk and magnetic tape is automatic error retry. Error processing for unit record devices except the system console is accomplished by IOCS typing the message INOP to the console, which allows the operator to retry or abort the I/O operation. If the operator aborts the I/O operation, or if automatic error retry for disk or magnetic tape is unsuccessful, an error status message is typed to the console and the error return address is taken if provided. Otherwise, the task is aborted.
2	DFI	data formatting is inhibited. Otherwise, data formatting is performed by the appropriate device handler. See Table L-1 for more explanation.
3	NST	device handlers perform no status checking and no status information is returned. All I/O appears to complete without error. Otherwise, status checking is performed and status information is returned as necessary.
4	RAN	file accessing occurs in the random mode. Otherwise, sequential accessing is performed. Note: This bit is set if word 2 bit 12 is set.
5		reserved (M.FILE)
6	EXP	must be 1 for 16-word FCB.
7	IEC	this bit is reserved for internal IOCS use.

Bits 8-12 Special Control Specification (FCB.SCFG). — This field contains device control specifications unique to certain devices. Interpretation and processing of these specifications are performed by the device handlers. A bit setting is meaningful only when a particular type of device is assigned as indicated in Table L-1.

Bits 13-31 reserved for extended control specifications

Bit	Meaning if Set	Definition
13	RXON	software read flow control required for 8-Line ACM (FCB.RXON)

Table L-1
Special Control Flags

Device	Bit 2=0	Bit 2=1	Bit 8=0	Bit 8=1	Bit 9=0	Bit 9=1
Line Printer (LP)	Interpret first character as carriage control	Interpret first character as data See bit 8	Form control	No form control		
Discs, (DM,DF, FL)	Report EOF if X'0FE0FE0F' encountered in word 0 of 1st block during read of unblocked file	X'0FE0FE0F' in word 0 not recognized as EOF				
8-Line Asynchronous Communications Multiplexer (TY)	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**
	Perform special character formatting	No special character formatting	ASCII control passed as data	ASCII control character detect	Echo by controller	No echo by controller
	M.WRIT	**M.WRIT**	**SVC 1,X'3E'**	**SVC 1,X'3E'**	**M.WRIT**	**M.WRIT**
	Interpret first character as carriage control	Interpret first character as data	Stop transmitting break	Start transmitting break	Normal write	Initialize device (load UART parameters)

Device	Bit 10=0	Bit 10=1	Bit 11=0	Bit 11=1	Bit 12=0	Bit 12=1
Line Printer (LP)	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Discs, (DM,DF, FL)					Normal read	Read with byte granularity (word 2 bit 4 set)
8-Line Asynchronous Communications Multiplexer (TY)	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**
	(If bit 2=0) convert lower case character to upper case	Inhibit conversion	No special character detect	Special character detect	Do not purge type ahead buffer	Purge type ahead buffer
	M.WRIT	**M.WRIT**	**M.WRIT**	**M.WRIT**	**M.WRIT**	**M.WRIT**
			Normal write	Write with input sub-channel monitoring plus software flow control		

Continued on next page

Table L-1
Special Control Flags (*Continued*)

Device	(Bit 2=0)	(Bit 2=1)	Bit 8	Bit 9		Bit 10	Bit 11	Bit 12
ALIM	Read:	**Bit 2**	**Bit 8**	**Bit 9**	**Read**	On Read:		
(Asynch-	receive							
ronous	data	0	1	0	=Blind mode reset	1= Inhibit		
Line	(bytes)	0	0	1	=Echo on read	conversion		
Interface	defined	1	N/A	N/A	=Receive data	of lower		
Module)	for	0	0	0	=Receive data	case		
Terminals	transfer					characters		
(TY)	count					to upper		
					Write	case		
	Write:	0	N/A	0	=Formatted write	0= Convert		
	formatted	0	N/A	1	=Initialize device			
		1	N/A	N/A	=Unformatted write			

Word 3

Bits 0-31 Status word (FCB.SFLG) — 32 indicator bits are set by IOCS to indicate the status, error, and abnormal conditions detected during the current or previous operation. The assignment of these bits is shown as follows:

Bits	Meaning if Set	Definition
0	OP	operation in progress. Request has been queued. (Note: Reset after post I/O processing complete.)
1	ERR	error condition found
2	BB	invalid blocking buffer control pointers have been encountered during file blocking or unblocking
3	PRO	write protect violation
4	INOP	device inoperable
5	BOM	beginning-of-medium (BOM) (load point) or illegal volume number (multivolume magnetic tape)
6	EOF	end-of-file
7	EOM	end-of-medium (end of tape, end of disk file)
8-9		reserved
10	TIME	last command exceeded time-out value and was terminated
11-15		reserved
16	ECHO	echo
17	INT	post program-controlled interrupt
18	LEN	incorrect length
19	PROG	channel program check
20	DATA	channel data check
21	CTRL	channel control check
22	INTF	interface check
23	CHAI	chaining check
24	BUSY	busy
25	ST	status modified
26	CTR	controller end
27	ATTN	attention
28	CHA	channel end
29	DEV	device end
30	CHK	unit check
31	EXC	unit exception

Word 4

Bits 0-31 Record length (FCB.RECL) — this field is set by IOCS to indicate the actual number of bytes transferred during read/write operations.

Word 5

Bits 0-7 Reserved

Bits 8-31 I/O queue address (FCB.IOQA) — this field is used by IOCS to point to the I/O queue for an I/O request initiated from this FCB

Word 6

Bits 0-7 Special status bits (FCB.SPST). The interpretation of these bits is shown below:

Bits	Definition
0	no-wait normal end action not taken
1	no-wait error end action not taken
2	request killed, I/O not issued
3	if set, exceptional condition has occurred in the I/O request
4	if set, software read flow control required
5-7	reserved

Bits 8-31 Wait I/O error return address (FCB.ERRT) — this field is set by the user and contains the address to which control is to be transferred in the case of an unrecoverable error when control bits 1 and 3 of word 2 are reset. If this field is not initialized and an unrecoverable error is detected under the above conditions, the requesting task is aborted.

Word 7

Bits 0-7 Index to FPT (FCB.FPTI) — this field is set by IOCS to index into the associated entry in the file pointer table (FPT)

Bits 8-31 FAT address (FCB.FATA) — this field is set by IOCS to point to the associated file assignment table (FAT) entry.

Word 8

Bits 0-7 Reserved

Bits 8-31 Data buffer address (FCB.ERWA) — start address of data area for read or write operations. (24 bit pure address)

Word 9

Bits 0-31 Quantity (FCB.EQTY)— number of bytes of data to be transferred

Word 10

Bits 0-31 Random access address (FCB.ERAA) — this field contains a block number (zero origin) relative to the beginning of the disk file. It is the start address for the current read or write operation with word 2 bit 4 set and word 2 bit 12 reset.

or

For disk read requests with word 2 bits 4 and 12 set (read with byte granularity), this word defines the byte offset relative to the beginning of the file. Note: If word 9 is zero, the file retains its position prior to the call.

Word 11

Bits 0-31 Status word one (FCB.IST1) — these are the first 32 bits of status returned by the sense command

Word 12

Bits 0-31 Status word two (FCB.IST2) — these are the second 32 bits of status returned by the sense command

Word 13

Bits 0-7 Reserved

Bits 8-31 No-wait I/O (FCB.NWOK) — normal completion return address. This user routine must be exited by calling the M.XIEA service.

Word 14

Bits 0-7 Reserved

Bits 8-31 No-wait I/O (FCB.NWER) — error completion return address. This user routine must be exited by calling the M.XIEA service.

Word 15 (Applicable only to volume resource.)

Bits 0-7 (FCB.BBN) — Number of 192 word buffers for user supplied blocking buffers. A value of one or zero in this field specifies one blocking buffer.

Bits 8-31 Blocking buffer address (FCB.BBA) — starting address of a contiguous area of memory FCB.BBN buffers long

Table L-2
Device Functions (Standard Devices)

Operation	IOCS Op Code	Line Printer (LP)	Mag Tape (M9/MT)	Disk (DM/DF/ DC/Floppy)	Handler=F8XIO (8-Line)
Open (M.FILE)	0	IOCS opens	IOCS opens	IOCS opens	Initialize IOP channel if necessary
Rewind (M.RWND)	1	Eject,set BOM bit word 3 bit 5 in FCB	Rewind Tape	Set current block address to zero (FAT)	SENSE operation
Read Record (M.READ)	2	Spec error	Read to data buffer	Read to data buffer	Read to data buffer
Write record (M.WRIT)	3	Write from data buffer	Write from data buffer. If blocked writes *n* data buffers to blocking buffer before output	Write from data buffer. If blocked IOCS writes *n* data buffers to blocking buffer before output	Write record to terminal
Write EOF (M.WEOF)	4	NOP*	Write EOF	If blocked, IOCS writes EOF. If unblocked writes X'0FE0FE0F'	NOP*
Execute Channel	5	Spec error	Execute Channel Program	Execute Channel Program	Execute channel Program

*NOP — No operation performed

Continued on next page

Table L-2
Device Functions (Standard Devices) *(Continued)*

Operation	IOCS Op Code	Line Printer (LP)	Mag Tape (M9/MT)	Disk (DM/DF/ DC/Floppy)	Handler=F8XIO (8-Line)
Advance Record (M.FWRD)	6	Spec error	Advance record	If blocked, advance record. If unblocked, advance one 192W block.	Set data terminal ready
Advance File (M.FWRD)	7	Spec error	Advance file (past EOF)	Spec error	Reset data terminal ready
Backspace Record (M.BACK)	8	Spec error	Backspace record	If blocked, backspace record. If unblocked backspace one 192W block	Used by J.TINIT to initialize terminals
Backspace File (M.BACK)	9	Spec error	Backspace file to previous EOF	Spec error	Reset request to send command
Upspace (M.UPSP)	A	Upspace	Multivolume only. If BOT, writes volume record. If EOT, performs ERASE, writes EOF, and issues MOUNT message.	Spec error on F-class disks. For floppy only: format diskette. New diskettes must be formatted prior to normal usage.	Set request to send command
Erase or Punch Trailer Not user IOCS/handler provides call automatically	B	NOP	Multivolume only. Same as upspace above. Erases 4" of tape before writing	NOP	Set/reset break (depends on flags in FCB)

Continued on next page

Table L-2
Device Functions (Standard Devices) *(Continued)*

Operation	IOCS Op Code	Line Printer (LP)	Mag Tape (M9/MT)	Disk (DM/DF/ DC/Floppy)	Handler=F8XIO (8-Line)
Eject/ Punch Leader (M.EJECT)	C	Eject to top of form	Write dummy record with eject control character as first character	NOP	Define special character
Close (M.CLSE)	D	IOCS closes	IOCS closes	IOCS closes	NOP
Reserve FHD Port	E	Spec error	Spec error	Reserve port-4MB disk only. Else, spec error Reserve Dual Ported Disk	Set single-channel operation (default) command
Release FHD Port	F	Spec error	Spec error	Release port-4MB disk only. Else, spec error Reserve Dual Ported Disk	Set dual-channel operation

Table L-3
Device Functions (Terminals, Handler Action Only)

Operation	IOCS Op Code	Handler = H.ASMP (ALIM)
Open M.FILE	0	NOP*
Rewind M.RWND	1	NOP*
Read record M.READ	2	Read to data buffer
Write record M.WRIT	3	Write record to terminal
Write EOF M.WEOF	4	NOP*
Execute channel	5	Execute channel
Advance record M.FWRD	6	Connect communications channel
Advance file M.FWRD	7	Disconnect communications channel
Backspace record M.BACK	8	Initialize device and set time-out value
Backspace file M.BACK	9	Clear break status flag word
Upspace M.UPSP	A	Spec error**
Erase/punch trailer	B	Transmit break
Eject/punch leader M.EJECT	C	Spec error**
Close M.CLSE	D	NOP*
Reserve FHD port	E	Spec error**
Release FHD port	F	Spec error**

* NOP = No operation performed

** Spec Error = Illegal operation code

Table L-4
Standard Carriage Control Characters and Interpretation

Control Character	Hexa-decimal Value	Result on a Terminal	Result on Directly Allocated Printer (Serial or parallel)	SLO
Blank	20	One linefeed, one carriage return before write	Single space before print	Single space before print
0	30	Two linefeeds, one carriage return before write	Double space before print	Double space before
1	31	Five linefeeds, one carriage return before write	Page eject (slew) before print	Page eject (slew) before print
+	2B	No linefeed, no carriage return before write (line append)	No space before print (overprint)	No space before print (overprint)
-	2D	Five linefeeds, one carriage return before write	Single space before print	Page eject, save and print up to three user supplied title lines. See Note 1.
<	3C	One linefeed, one carriage return before write	Single space before print	Set inhibit spooler title line in this file.
>	3E	One linefeed, one carriage return before write	Single space before print	Set enable spooler title line in this file.
=	3D	One linefeed, one carriage return before write	Single space before print	Page eject and clear up to three user-supplied title lines in this file.

Notes:

1. User-supplied title lines have the same effect as this character. Supplying a fourth title line clears the first three, but only one page is ejected. User-supplied titles are retained by the spooler and are repeated at the top of each page until cleared or the spool file ends.

L.6 File Control Block, Compatible Mode 8 word (FCB)

Word	0–7	8–12	13–31
0	Opcode (FCB.OPCD)	Logical file code (FCB.LFC)	
1	Transfer control word (FCB.TCW)		
2	General control flags (FCB.GCFG)	Special flags (FCB.SCFG)	Random access address (FCB.CBRA)
3	Status flags (FCB.SFLG)		
4	Actual transfer quantity (FCB.RECL)		
5	Reserved	I/O queue address (FCB.IOQA)	
6	Special Status (FCB.SPST)	Wait I/O error return address (FCB.ERRT)	
7	Index to FPT (FCB.FPTI)	FAT address (FCB.FATA)	

Shaded areas are set by the system.

A.L8W.FCB

Word 0

Bit 0 Reserved

Bits 1-7 Operation code (FCB.OPCD) — type of function requested of the device handler. This field is set by IOCS as a function of the requested service.

Bits 8-31 Logical file code (FCB.LFC) — any combination of three ASCII characters is allowed.

Word 1 (FCB.TCW)

This word supplies a transfer control word (TCW) that describes a data buffer and transfer quantity. If no TCW definition is supplied, the transfer buffer defaults to location zero of the task's logical address space and is 4096 words long.

Bits 0-11 Quantity — 12 bit field specifying the number of data items to be transferred. This quantity must include the carriage control character, if applicable. The transfer quantity is in units determined by the address in bits 12 to 31.

Bits 12-31 Format code and buffer address— bits 12, 30 and 31 specify byte, halfword, or word quantities for data transfers. They are interpreted as follows:

Type of Transfer	F (12)	C (30,31)	Address
Byte	1	xx	13-31
Halfword	0	x1	13-30
Word	0	00	13-29

Word 2

Bits 0-7 General control flags (FCB.GCFG) — these eight bits enable the user to specify the manner in which an operation is to be performed by IOCS. The interpretation of these bits is shown below:

File Control Block, Compatible Mode 8 word (FCB)

Bit	Meaning if Set	Definition
0	NWT	IOCS returns to the user immediately after the I/O operation is queued. If reset, IOCS exits to the calling program only when the requested operation has been completed.
1	NER	error processing is not performed by either the device handler or IOCS. An error return address is ignored and a normal return is taken to the caller; however, the device status is posted in the FCB unless bit 3 is set. If reset, normal error recovery is attempted. Normal error processing for disk and magnetic tape is automatic error retry. Error processing for unit record devices except the system console is accomplished by IOCS typing the message INOP to the console, which allows the operator to retry or abort the I/O operation. If the operator aborts the I/O operation, or if automatic error retry for disk or magnetic tape is unsuccessful, an error status message is typed to the console and the error return address is taken if provided. Otherwise, the task is aborted.
2	DFI	data formatting is inhibited. Otherwise, data formatting is performed by the appropriate device handler. See Table L-5 for more explanation.
3	NST	device handlers perform no status checking and no status information is returned. All I/O appears to complete without error. Otherwise, status checking is performed and status information is returned as necessary.
4	RAN	file accessing occurs in the random mode. Otherwise, sequential accessing is performed.
5		reserved (M.FILE)
6	EXP	must be 0 for 8 word FCB.
7	IEC	this bit is reserved for internal IOCS use.

Bits 8-12 Special Control Specification (FCB.SCFG). — This field contains device control specifications unique to certain devices. Interpretation and processing of these specifications are performed by the device handlers. A bit setting is meaningful only when a particular type of device is assigned as indicated in Table L-2.

Bits 13-31 Random access address (FCB.CBRA) — This field contains a block number (zero origin) relative to the beginning of the disk file, and specifies the base address for read or write operations.

Table L-5
Special Control Flags (8 Word FCB)

Device	Bit 2=0	Bit 2=1	Bit 8=0	Bit 8=1	Bit 9=0	Bit 9=1
Line Printer (LP)	Interpret first character as carriage control	Interpret first character as data See bit 8	Form control	No form control		
Discs, (DM,DF, FL)	Report EOF if X'0FE0FE0F' encountered in word 0 of 1st block during read of unblocked file	X'0FE0FE0F' in word 0 not recognized as EOF				
8-Line Asynchronous Communications Multiplexer (TY)	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**
	Perform special character formatting	No special character formatting	ASCII control passed as data	ASCII control character detect	Echo by controller	No echo by controller
	M.WRIT	**M.WRIT**	**SVC 1,X'3E'**	**SVC 1,X'3E'**	**M.WRIT**	**M.WRIT**
	Interpret first character as carriage control	Interpret first character as data	Stop transmitting break	Start transmitting break	Normal write	Initialize device (load UART parameters)

Device	Bit 10=0	Bit 10=1	Bit 11=0	Bit 11=1	Bit 12=0	Bit 12=1
Line Printer (LP)	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Discs, (DM,DF, FL)						
8-Line Asynchronous Communications Multiplexer (TY)	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**	**M.READ**
	(If bit 2=0) convert lower case character to upper case	Inhibit conversion	No special character detect	Special character detect	Do not purge type ahead buffer	Purge type ahead buffer
	M.WRIT	**M.WRIT**	**M.WRIT**	**M.WRIT**	**M.WRIT**	**M.WRIT**
			Normal write	Write with input sub-channel monitoring plus software flow control		

Continued on next page

Table L-5
Special Control Flags (8 Word FCB) (*Continued*)

Device	(Bit 2=0)	(Bit 2=1)	Bit 8	Bit 9		Bit 10	Bit 11	Bit 12
ALIM	Read:	**Bit 2**	**Bit 8**	**Bit 9**	**Read**	On Read:		
(Asynch-	receive							
ronous	data	0	1	0	=Blind mode reset	1= Inhibit		
Line	(bytes)	0	0	1	=Echo on read	conversion		
Interface	defined	1	N/A	N/A	=Receive data	of lower		
Module)	for	0	0	0	=Receive data	case		
Terminals	transfer					characters		
(TY)	count					to upper		
					Write	case		
	Write:	0	N/A	0	=Formatted write	0= Convert		
	formatted	0	N/A	1	=Initialize device			
		1	N/A	N/A	=Unformatted write			

Word 3

Bits 0-31 Status word (FCB.SFLG) — 32 indicator bits are set by IOCS to indicate the status, error, and abnormal conditions detected during the current or previous operation. The assignment of these bits is shown as follows:

Bits	Meaning if Set	Definition
0	OP	operation in progress. Request has been queued. (Note: Reset after post I/O processing complete.)
1	ERR	error condition found
2	BB	invalid blocking buffer control pointers have been encountered during file blocking or unblocking
3	PRO	write protect violation
4	INOP	device inoperable
5	BOM	beginning-of-medium (BOM) (load point) or illegal volume number (multivolume magnetic tape)
6	EOF	end-of-file
7	EOM	end-of-medium (end of tape, end of disk file)
8-9		reserved
10	TIME	last command exceeded time-out value and was terminated
11-15		reserved
16	ECHO	echo
17	INT	post program-controlled interrupt
18	LEN	incorrect length
19	PROG	channel program check
20	DATA	channel data check
21	CTRL	channel control check
22	INTF	interface check
23	CHAI	chaining check
24	BUSY	busy
25	ST	status modified
26	CTR	controller end
27	ATTN	attention
28	CHA	channel end
29	DEV	device end
30	CHK	unit check
31	EXC	unit exception

Word 4

Bits 0-31 Record length (FCB.RECL) — this field is set by IOCS to indicate the actual number of bytes transferred during read/write operations.

Word 5

Bits 0-7 Reserved

Bits 8-31 I/O queue address (FCB.IOQA) — this field is used by IOCS to point to the I/O queue for an I/O request initiated from this FCB

Word 6

Bits 0-7 Special status bits (FCB.SPST). The interpretation of these bits is shown below:

Bits	Definition
0	no-wait normal end action not taken
1	no-wait error end action not taken
2	kill command, I/O not issued
3	if set, exceptional condition has occurred in the I/O request
4	if set, software read flow control required
5-7	reserved

Bits 8-31 Wait I/O error return address (FCB.ERRT) — this field is set by the user and contains the address to which control is to be transferred in the case of an unrecoverable error when control bits 1 and 3 of word 2 are reset. If this field is not initialized and an unrecoverable error is detected under the above conditions, the user is aborted.

Word 7

Bits 0-7 Index to FPT (FCB.FPTI) — this field indexes into the appropriate entry in the file pointer table (FPT)

Bits 8-31 FAT address (FCB.FATA) — this field points to the file assignment table (FAT) entry associated with all I/O performed for this FCB. This field is supplied by IOCS.

L.7 File Control Block (FCB), High Speed Data

The following section details the 16 words that make up the FCB for the HSD.

Word	0 – 7	8 – 15	16 – 23	24 – 31
0	Opcode (FCB.OPCD)	Logical file code (FCB.LFC)		
1	Reserved			
2	General control flags (FCB.GCFG)	Special flags (FCB.SCFG)	Reserved	UDDCMD of IOCD if bit 11 of word 2 is set
3	Status flags (FCB.SFLG)			
4	Record Length in bytes (FCB.RECL)			
5	Reserved	I/O queue address (FCB.IOQA)		
6	Special Status (FCB.SPST)	Wait I/O error return address (FCB.ERRT)		
7	Index to FPT (FCB.FPTI)	FAT address (FCB.FATA)		
8	Reserved	Data address (FCB.ERWA)		
9	Transfer quantity (bytes) (FCB.EQTY)			
10	Device command for non-EXCPM (FCB.ERAA)			
11	Reserved			
12	Extended I/O status word two (FCB.IST2)			
13	Reserved	No-wait I/O normal end-action service address (FCB.NWOK)		
14	Reserved	No-wait I/O error end-action service address (FCB.NWER)		
15	Reserved			

Shaded areas are set by the system.

T2FCB

Word 0

Bit 0 Reserved

Bits 1-7 Contain the operation code, set by IOCS that specifies the type of function requested of H.HSDG.

Bits 8-31 Contain the logical file code associated with the device for the I/O operation.

Word 1

This word is reserved and should be set to zero.

Word 2

Bits 0-7 Contain control flags that enable the user to specify how an operation is to be performed by IOCS. Following is the meaning of these bits when they are set:

Bit	Meaning When Set
0	IOCS returns to the user immediately after the I/O operation is queued (no wait I/O). If reset, IOCS exits to the calling program only when the HSD completes the requested operation (wait I/O).
1	H.HSDG and IOCS do not perform error processing. IOCS ignores the error return address and takes a normal return to the caller. H.HSDG posts device status in the FCB (unless bit 3 is set). If reset, H.HSDG and IOCS perform error processing.
2	specifies physical execute channel program. If reset, specifies logical channel program or non-execute channel program I/O request.
3	IOCS performs no status checking and does not return status information. All I/O appears to complete without error. If reset, IOCS performs status checking and returns status information.
4, 5	Reserved, should be zero.
6	specifies 16 word FCB. Must be set to 1.
7	reserved for internal IOCS use.

Bits 8-23 contain the following special flags:

Bit	Meaning When Set
8	specifies request device status after a transfer. H.HSDG adds an IOCB to the IOCL to retrieve device-specific status after the data transfer completes.
9	specifies send device command prior to data transfer. H.HSDG prefixes the transfer with an IOCB that sends a device command word to the device. The value sent is the 32-bits contained in word 10 of the FCB.
10	specifies disable time out for this request. This bit indicates the operation will take an indeterminable period of time. In most cases this applies only to read operations.
11	specifies set UDDCMD from the least significant byte of word 2. This bit indicates that the UDDCMD byte in the data transfer IOCB must be set to the least significant byte of FCB word 2. This allows the user to pass additional control information to the device without modifying the device driver.
12	specifies disable asynchronous status notification during no-wait I/O.
13	specifies the execute channel program request INIT. By setting this bit, all preliminary I/O data structures are set up based on the I/O command list address provided in word 8 of the FCB. When set, this bit prepares for future cyclic I/O requests but does not issue any I/O.
14	specifies the execute channel program request GO. This bit issues an SIO for the most recently processed INIT execute channel program request (see bit 13).
15-23	reserved
Note:	For further information on the HSD FCB please see the H.HSDG section in the MPX-32 Technical Manual Volume II.

Bits 24-31 if bit 11 is set, these bits define the UDDCMD field of the generated IOCB, overriding the default value from a handler table. This field applies only to FCB format.

Word 3

IOCS uses this word to indicate status, error, and abnormal conditions detected during the current or previous operation. Following is the meaning of the bits when they are set:

Bit	Meaning When Set
0	operation in progress. Request has been queued. This bit is reset after post I/O processing completes.
1	error condition found
2, 3	not applicable, should never be returned
4	device inoperable, HSD not present or offline
5-15	not applicable, should never be returned
16	a time-out occurred and a CD terminate was issued.
17, 18	not applicable, should never be returned
19	there was data remaining in the HSD fifo when the transfer count equaled zero.
20	a parity error occurred during the current data transfer.
21	a non-present memory error occurred during the current data transfer.
22	program violation. An invalid operation code was detected.
23	device inoperative
24	HSD data buffer overflow. Some data from the device was lost.
25	external termination
26	IOCB address error
27	error on TI address fetch
28	device EOB
29	Non-device access errors precluded request queuing. For a list of the errors, see word 12.
30, 31	non-execute channel program type of IOCB in error as follows:

Value	Meaning
00	data transfer
01	device status
10	command transfer

Word 4

This word specifies the record length. For non-execute channel program I/O, IOCS sets this word to indicate the number of bytes transferred during read or write operations.

Word 5

Bits 0-7 reserved

Bits 8-31 specify the IOQ address. IOCS sets this field to point to the IOQ entry initiated from this FCB.

Word 6

Bits 0-7 specify special status as follows:

Bit	Meaning When Set
0	no-wait normal end action not taken
1	no-wait error end action not taken
2	kill command, I/O not issued
3	an exception condition has occurred in the I/O request
4	not used
5-7	reserved

Bits 8-31 contain the wait I/O error return address. The user sets this field to the address where control is to be transferred for unrecoverable errors when bits 0, 1, and 3 of word 2 are reset. If this field is not initialized and an unrecoverable error is detected under the above conditions, the user task is aborted.

Word 7

Bits 0-7 set by the I/O control system (IOCS), contains an index to the file pointer table (FPT) entry for this I/O operation.

Bits 8-15 supplied by the IOCS, points to the file assignment table (FAT) entry associated with this FCB.

Word 8

Bits 0-7 reserved

Bits 8-31 these bits are used as the data address, a logical IOCL address, or a physical IOCL address as follows:

Data address – This is the starting address of the data area for FCB format I/O operations. This address must be a word address.

Logical IOCL address – This is a logical, doubleword address that points to a user-supplied IOCL for SIO format I/O operations. For more information about SIO format, refer to Reference Manual Volume I, Chapter 3. The execute channel program entry point (H.IOCS,10) must be used and bit 2 of word 2 of the FCB is reset. All addresses within the IOCL are assumed to be logical and map block boundary crossings need not be resolved.

Physical IOCL address – This is a physical, doubleword address that points to a user-supplied IOCL for SIO format I/O operations. The execute channel program entry point (H.IOCS,10) must be used and bit 2 of word 2 of the FCB is set. All addresses within the IOCL are assumed to be physical and all map block boundary crossings are assumed to be resolved.

Word 9

This word specifies the number of bytes of data to be transferred.

Word 10

For nonexecute channel program format, this word defines a device command.

Word 11

Reserved — should be set to zero.

Word 12

This word contains status sent from the user's device or if bit 29 of word 3 is set, this word defines the opcode processor (EP5) detected errors as follows:

Value	Explanation
1	request made with non-expanded FCB
2	FCB format transfer count was zero
3	FCB format, byte transfer count was not a multiple of 4 bytes
4	SIO format with a physical IOCL request by an unprivileged caller
5	SIO format with a physical IOCL request by a nonresident caller
6	first IOCB in caller's IOCL is a transfer in channel
7	caller's IOCL not on a doubleword boundary
8	SIO format IOCL contains an IOCB with a zero transfer count
9	infinite transfer in channel loop
10	consecutive SOBNZ's in IOCL

11	SOBNZ target is not in the IOCL
12	the transfer address is not on a word boundary
13	unprivileged caller's input buffer includes protected locations
14	unprivileged caller's input buffer is unmapped either in MPX-32 or below DSECT
15	cyclic I/O request was made for which no cyclic IOQ is current
16	cyclic I/O request was made and permanent IOQ support was not sysgened into the system

Word 13

Bits 0-7 reserved

Bits 8-31 contain the address of the user-supplied routine to branch to for no-wait I/O normal completion. This routine must be terminated by calling H.IOCS,34 (no-wait I/O end action return). If word 2 bit 12 is reset, this address plus one word is the location where control is transferred on asynchronous status notification.

Word 14

Bits 0-7 reserved

Bits 8-31 contain the address of the user-supplied routine to branch to for no-wait I/O error completion. This routine must be terminated by calling H.IOCS,34 (no-wait I/O end action return).

Word 15

Reserved — should be set to zero.

L.8 File Pointer Table (FPT)

The file pointer table (FPT) provides the linkage between the file control block (FCB) and the file assignment table (FAT). It also allows for multiple logical file code assignments to be made equivalent to the same FAT. The linkage to the FAT is performed at assignment. The linkage to the FCB is performed at open and is re-established if necessary for every operation at opcode processing time. The FPT resides in the task's service area.

FPT entries one to six are reserved for the system as follows:

Entry 1 - System LFC *s*
Entry 2 - Load module LFC *LM
Entry 3 - H.VOMM resource descriptor LFC (1)
Entry 4 - H.VOMM directory LFC (2)
Entry 5 - H.VOMM DMAP/SMAP LFC (3)
Entry 6 - H.VOMM modify resource descriptor LFC X'FFFEE'

Each FPT entry has the following format:

	0 ... 7	8 ... 15 16 ... 23 24 ... 31
Word 0	Reserved	Logical file code (FPT.LFC)
1	Flags (FPT.FLGS). See Note 1.	FCB address (FPT.FCBA)
2	Reserved	FAT address (FPT.FATA)

Notes:

1. Bits in FPT.FLGS are assigned as follows:

Bit	Meaning if Set
0	reserved
1	multiple FPT entries exist that point to the same FAT (i.e., $ASSIGN4 or $ASSIGN *lfc* TO LFC = *lfc* statements)
2	FPT busy flag
3	FPT open
4	this FPT entry is not in use
5	pseudo-SYC assignment (used by TSM)
6	pseudo-FPT for unassigned tempory file
7	reserved

L.9 Parameter Task Activation Block

The following is the structure of the expanded parameter task activation block:

Byte	Word	0–7	8–15	16–23	24–31
0	0	PTA.FLAG	PTA.NRRS	PTA.ALLO	PTA.MEMS
4	1	PTA.NBUF	PTA.NFIL	PTA.PRIO	PTA.SEGS
8	2-3	PTA.NAME			
10	4-5	PTA.PSN			
18	6-7	PTA.ON			
20	8-9	PTA.PROJ			
28	10	PTA.VAT	PTA.FLG2	PTA.EXTD	
2C	11	PTA.PGOW			
30	12	PTA.TSW			
34	13	PTA.RPTR			
38	14	PTA.PGO2			
3C	15	PTA.FSIZ		PTA.RSIZ	
40	16-19	Reserved (zero)			
50-*nn*	20-*nn*	RRS List			

Byte (Hex)	Symbol	Description
0	PTA.FLAG	contains the following:

Bit	Contents
0	reserved
1	job oriented (PTA.JOB)
2	terminal task (PTA.TERM)
3	batch task (PTA.BTCH)
4	debug overlay required (PTA.DOLY)
5	resident (PTA.RESD)
6	directive file active (PTA.DFIL)
7	SLO assigned to SYC (PTA.SLO)

For unprivileged callers, bits 0-3 are not applicable. These characteristics are inherited from the parent task.

Byte (Hex)	Symbol	Description
1	PTA.NRRS	number of resource requirements or zero if same as summary entries in the load module or executable image preamble

Byte (Hex)	Symbol	Description
2	PTA.ALLO	memory requirement: number of 512-word pages exclusive of TSA, or zero if same as the preamble
3	PTA.MEMS	memory class (ASCII E, H or S) or zero if memory class is to be taken from the preamble. If the memory class is to be taken from the preamble, the caller has the option of specifying the task's logical address space in this field as follows: Bits — Contents 0-3 — hexadecimal value 0 through F representing the task's logical address space in megabytes where zero is 1MB and F is 16MB 4-7 — zero
4	PTA.NBUF	the number of blocking buffers required or zero if same as the preamble
5	PTA.NFIL	the number of FAT/FPT pairs to be reserved or zero if same as the preamble
6	PTA.PRIO	the priority level at which the task is to be activated or zero for the cataloged load module priority. See the Parameter Send Block section in Chapter 2 of the MPX-32 Reference Manual Volume I, for more details.
7	PTA.SEGS	the segment definition count or reserved (zero)
8	PTA.NAME	contains the load module or executable image name, left justified and blank filled, or word 2 is zero and word 3 contains a pathname vector or RID vector
10	PTA.PSN	contains the 1- to 8-character ASCII pseudonym, left justified and blank filled, to be associated with the task or zero if no pseudonym is desired. For unprivileged callers, this attribute is inherited from the parent task if zero is supplied or the parent is in a terminal or batch job environment.
18	PTA.ON	contains the 1- to 8-character ASCII owner name, left-justified and blank-filled, to be associated with the task or zero if the task to default to the current owner name. Valid only when task has system administrator attribute.
20	PTA.PROJ	contains the 1- to 8-character ASCII project name, left-justified and blank-filled, to be associated with files referenced by this task, or zero if same as LMIT
28	PTA.VAT	the number of volume assignment table (VAT) entries to reserve for dynamic mount requests or zero if same as the preamble

Byte (Hex)	Symbol	Description
29	PTA.FLG2	contains the following flags: Bit — Meaning if Set 0 — debug activating task (PTA.DBUG) 1 — Command Line Recall and Edit is in effect for the task (PTA.CLRE) 2 — NOTSA option (PTA.NTSA) 3 — TSA option (PTA.TSA) 4 — expanded PTASK block flag (must be set to use options 33-64) (PTA.EBLK) 5 — reserved (zero) 6 — enables NOMAPOUT option (PTA.NMAP) 7 — enables MAPOUT option (PTA.MAP)
2A	PTA.EXTD	contains the following values: Bit — Meaning if Set -1 — maxaddr of extended MPX-32 and TSA -2 — minaddr of extended MPX-32 and TSA 0 — invalid with PTA.TSA or PTA.NTSA option *n* — a positive number representing a map block of MPX-32 and TSA
2C	PTA.PGOW	contains the initial value of the task option word or zero
30	PTA.TSW	contains the initial value of the task status word or zero
34	PTA.RPTR	contains a pointer to the resource requirement summary list or, if an expanded PTASK block is not used, the RRS list begins here (see resource requirement summary list description below)
38	PTA.PGO2	contains the initial value of the second task option word
3C	PTA.FSIZ	contains the length of the fixed portion of the PTASK block in bytes
3E	PTA.RSIZ	contains the number of bytes of the resource requirement summary
40	Reserved	

Byte (Hex)	Symbol	Description
50		resource requirement summary list. Each entry contains a variable length RRS. The RRS list has up to 384 words. Each entry must be doubleword bounded. Each entry is compared with the RRS entries in the LMIT. If the logical file code currently exists, the specified LFC assignment will override the cataloged assignment, otherwise the special assignment will be treated as an additional requirement and merged into the list. If MPX-32 Revision 1.*x* format of the RRS is specified, it is converted to the format acceptable for assignment processing by the Resource Management Module (H.REMM). See MPX-32 Revision 1.*x* Technical Manual for format of the RRS.

L.10 TSM Procedure Call Block (PCB)

The PCB contains the information necessary for the service to complete a procedure call. The format of the PCB is as follows:

<table>
<tr><th></th><th>0 … 7</th><th>8 … 15</th><th>16 … 23</th><th>24 … 31</th></tr>
<tr><td>Word 0</td><td></td><td colspan="3">Send buffer address (PCB.SBA)</td></tr>
<tr><td>1</td><td colspan="4">Send quantity (PCB.SQUA)</td></tr>
<tr><td>2</td><td></td><td colspan="3">Return buffer address (PCB.RBA)</td></tr>
<tr><td>3</td><td colspan="2">Actual return length (PCB.ACRP)</td><td colspan="2">Return buffer length (PCB.RPBL)</td></tr>
</table>

Send buffer address	is the address of a character string that represents a valid TSM procedure call directive
Send quantity	contains the length in bytes of the TSM procedure call directive
Return buffer address	is the address of a buffer to contain either valid return information or an error message if CC1 is set and R7 contains a value of 1
Actual return length	is the number of bytes returned from the procedure call
Return buffer length	is the size in bytes of the supplied return buffer

L.11 Pathname Blocks (PNB)

The pathname block (PNB) is an alternative form of a pathname that can be used interchangeably with pathnames. Because of its structure, it can be parsed faster than a pathname. The PNB is a doubleword bounded, variable length ASCII character string which H.VOMM can distinguish from a pathname since the PNB always starts with an exclamation point.

H.VOMM provides a service to convert a pathname to a PNB. The examples which follow illustrate common pathnames and their corresponding PNB.

Example 1

`@VOL1(DIR1)FILE1`

Word	Contents
0	! V O L
1	blank
2	V O L 1
3	blank
4	blank
5	blank
6	! D I R
7	R O O T
8	D I R 1
9	blank
10	blank
11	blank
12	! R E S
13	blank
14	F I L E
15	1 b̸ b̸ b̸
16	blank
17	blank

Example 2

`FILE1`

Word	Contents
0	! V O L
1	W O R K
2	! D I R
3	W O R K
4	! R E S
5	blank
6	F I L E
7	1 b̸ b̸ b̸
8	blank
9	blank

Example 3

`(DIRECTORY)MYFILE`

Word	Contents
0	! V O L
1	W O R K
2	! D I R
3	R O O T
4	D I R E
5	C T O R
6	Y ƀ ƀ ƀ
7	blank
8	! R E S
9	blank
10	M Y F I
11	L E ƀ ƀ
12	blank
13	blank

Example 4

`@SYSTEM(SYSTEM)LOADMOD`

Word	Contents
0	! V O L
1	S Y S T
2	! D I R
3	S Y S T
4	! R E S
5	blank
6	L O A D
7	M O D ƀ
8	blank
9	blank

L.12 Post Program-Controlled Interrupt Notification Packet (PPCI)

If a task sets up a PPCI end-action receiver to check status during execution of its channel program, the status is returned in a notification packet. The address of the notification packet is contained in register three upon entering the task's PPCI end-action receiver. The notification packet is described below.

Word	0 7	8 15	16 23	24 31
Word 0	String forward address (NOT.SFA)			
1	String backward address (NOT.SBA)			
2	Link priority (NOT.PRI)	NOT.TYPE See Note 1.	Reserved	
3	FCB address (NOT.CODE)			
4	PSD 1 of task's PPCI receiver (NOT.PSD1)			
5	PSD 2 of task's PPCI receiver (NOT.PSD2)			
6	Number of PPCIs received since last buffer clear (NOT.STAR)		Number of status doublewords in status buffer (NOT.STAS)	
7	Address of PPCI status buffer (NOT.STAA)			
8	Address of buffer storing next status doubleword (NOT.STPT)			
9	Reserved			
10-*n*	PPCI status buffer			

Notes:

1. NOT.TYPE - Set to 1 for asynchronous notification.
2. Words 0-9 are updated by the operating system and must not be changed by the user.

L.13 Parameter Receive Block (PRB)

The parameter receive block (PRB) is used to control the storage of passed parameters into the receiver buffer of the destination task. The same format PRB is used for message and run requests. The address of the PRB must be presented when the M.GMSGP or M.GRUNP services are invoked by the receiving task.

Word	0 – 7	8 – 15	16 – 23	24 – 31
0	Status (PRB.ST)	Parameter receiver buffer address (PRB.RBA)		
1	Receiver buffer length (PRB.RBL)		Number of bytes actually received (PRB.ARQ)	
2	Owner name of sending task, word one (PRB.OWN)			
3	Owner name of sending task, word two			
4	Task number of sending task (PRB.TSKN)			

Notes:

1. Status (PRS.ST) contains the status-value encoded status byte:

Code	Definition
0	normal status
1	invalid PRB address (PRB.ER01)
2	invalid receiver buffer address or size detected during parameter validation (PRB.RBAE)
3	no active send request (PRB.NSRE)
4	receiver buffer length exceeded (PRB.RBLE)

2. Parameter receiver buffer address (PRB.RBA) contains the word address of the buffer where the sent parameters are stored.
3. Receiver buffer length (PRB.RBL) contains the length of the receiver buffer (0 to 768 bytes).
4. Number of bytes received (PRB.ARQ) is set by the operating system and is clamped to a maximum equal to the receiver buffer length.
5. Owner name of sending task (PRB.OWN) is a doubleword that is set by the operating system to contain the owner name of the task that issued the parameter send request.
6. Task number of sending task (PRB.TSKN) is set by the operating system to contain the task activation sequence number of the task that issued the parameter send request.

L.14 Parameter Send Block (PSB)

The parameter send block (PSB) describes a send request issued from one task to another. The same PSB format is used for both message and run requests. The address of the PSB (word bounded) must be specified when invoking the M.SMSGR or M.SRUNR services, but is optional when invoking the M.PTSK service.

When a load module name is supplied in words 0 and 1 of the PSB, the operating system searches the system directory only. For activations in directories other than the system directory, a pathname or RID vector must be supplied.

When activating a task with the M.SRUNR or M.PTSK service, the value specified in byte 0 of PSB word 2 (PSB.PRI) is used to determine the task's execution priority. This value overrides the cataloged priorities of the sending and receiving tasks and the priority specified in the PTASK block. However, priority clamping is used to prevent time-distribution tasks from using this value to execute at a real-time priority, and real-time tasks from executing at a time-distribution priority. Values that can be specified in PSB.PRI are 1-64 (to be the task priority), zero (to use the base priority of the sending task), and X'FF' (to ignore the PSB priority field).

A PSB can be specified as a parameter for the M.PTSK service, along with the required task activation (PTASK) block. The PTASK block also contains a priority specification field. The PSB priority value always overrides the PTASK block priority value.

	0 7	8 15	16 23	24 31
Word 0	Load module or executable image name (PSB.LMN) or zero if activation (or task number (PSB.TSKN) if message or run request to multicopied task)			
1	Load module or executable image name, pathname vector, or RID vector if activation (or zero if message or run request to multicopied task)			
2	Priority (PSB.PRI)	Reserved	Number of bytes to be sent (PSB.SQUA)	
3	Reserved	Send buffer address (PSB.SBA)		
4	Return parameter buffer length in bytes (PSB.RPBL)		Number of bytes actually returned (PSB.ACRP)	
5	Reserved	Return parameter buffer address (PSB.RBA)		
6	Reserved	No-wait request end action address (PSB.EAA)		
7	Completion status (PSB.CST)	Processing start status (PSB.IST)	User status (PSB.UST)	Options (PSB.OPT)

Word 0

Bits 0-31 — Load module or executable image name — contains characters 1 through 4 of the name of the load module or executable image to receive the run request or

Task number — contains the task number of the task to receive the message or the task number of the multicopied load module or executable image to receive the run request.

Word 1

Bits 0-31 — Load module or executable image name — contains characters 5 through 8 of the name of the load module or executable image to receive the run request, or zero if the message or run request is sent to multicopied load module or executable image.

Word 2

Bits 0-7 — Contains the priority at which the receiver task is expected to be activated. Valid values are 1-64, zero, (for base priority of the sending task) and X'FF', which generates activation priority based on a combination of values that can be specified during task activation.

The following tables show how the priority of a receiver task is determined when activated with M.SRUNR or with M.PTSK.

When Activating with M.SRUNR

Cataloged Priority of Send Task	Cataloged Priority of Receive task	Priority in PSB	Activates Receive task at
1-54	1-54	0	Send task cat. priority
1-54	55-64	0	55 (time-dist. clamp)
55-64	1-54	0	54 (real-time clamp)
55-64	55-64	0	Send task cat. priority
*	1-54	1-54	PSB priority
*	1-54	55-64	54 (real-time clamp)
*	55-64	1-54	55 (time-dist. clamp)
*	55-64	55-64	PSB priority
*	*	X'FF'	Receive task cat. priority

* not specified

When Activating with M.PTSK

Cataloged Priority of Send Task	Receive task	Priority in PTASK block	PSB	Activates Receive task at
1-54	1-54	0	0	Send task cat. priority
1-54	55-64	0	0	55 (time-dist. clamp)
1-54	*	1-54	0	Send task cat. priority
1-54	*	55-64	0	55 (time-dist. clamp)
55-64	1-54	0	0	54 (real-time clamp)
55-64	55-64	0	0	Send task cat. priority
55-64	*	1-54	0	54(real-time clamp)
55-54	*	55-64	0	Send task cat. priority
*	1-54	0	1-54	PSB priority
*	1-54	0	55-64	54 (real-time clamp)
*	55-64	0	1-54	55 (time-dist.clamp)
*	55-64	0	55-64	PSB priority
*	*	1-54	1-54	PSB priority
*	*	1-54	55-64	54 (real-time clamp)
*	*	1-54	X'FF'	PTASK block priority
*	*	55-64	1-54	55 (real-time clamp)
*	*	55-64	55-64	PSB priority
*	*	55-64	X'FF'	PTASK block priority
*	*	0	X'FF'	Receive task cat. priority

* not specified

Bits 8-15 reserved

Bits 16-31 Number of bytes to be sent — specifies the number of bytes to be passed (0 to 768) with the message or run request.

Word 3

Bits 0-7 reserved

Bits 8-31 Send buffer address — contains the word address of the buffer containing the parameters to be sent.

Word 4

Bits 0-15 Return parameter buffer length — contains the maximum number of bytes (0 to 768) that may be accepted as returned parameters.

Bits 16-31 Number of bytes actually returned — set by the send message or run request service upon completion of the request.

Word 5

Bits 0-7 reserved

Bits 8-31 Return parameter buffer address — contains the word address of the buffer where any returned parameters are stored.

Word 6

Bits 0-7 reserved

Bits 8-31 No-wait request end-action address — contains the address of a user routine to be executed at a software interrupt level upon completion of the request.

Word 7

Bits 0-7 Completion status — contains completion status information posted by the operating system as follows:

Bit	Meaning if Set
0	operation in progress (PSB.OIP)
1	destination task was aborted before completion of processing for this request (PSB.DTA)
2	destination task was deleted before completion of processing for this task (PSB.DTD)
3	return parameters truncated — attempted return exceeds return parameter buffer length (PSB.RPT)
4	send parameters truncated — attempted send exceeds destination task receiver buffer length (PSB.SPT)
5	user end-action routine not executed because of task abort outstanding for this task (can be examined in abort receiver to determine incomplete operation) (PSB.EANP)
6-7	reserved

Bits 8-15 — Processing start (initial) status — contains initial status information posted by the operating system as follows:

Bit	Meaning if Set
0	normal initial status (PSB.IST)
1	message request task number invalid (PSB.TSKE)
2	run request load module or executable image name not found (PSB.LMNE)
3	reserved
4	file associated with run request load module or executable image name does not have a valid load module or executable image format (PSB.LMFE)
5	dispatch queue entry (DQE) space is unavailable for activation of the load module or executable image specified by a run request (PSB.DQEE)
6	an I/O error was encountered while reading the directory to obtain the file definition of the load module or executable image specified in a run request (PSB.SMIO)
7	an I/O error was encountered while reading the file containing the load module or executable image specified in a run request (PSB.LMIO)
8	memory unavailable
9	invalid task number for run request to module or executable image in RUNW state
10	invalid priority specification. An unprivileged task can not specify a priority which is higher than its own execution priority (PSB.PRIE).
11	invalid send buffer address or size (PSB.SBAE)
12	invalid return buffer address or size (PSB.RBAE)
13	invalid no-wait mode end action routine address (PSB.EAE)
14	memory pool unavailable (PSB.MPE)
15	destination task receiver queue is full (PSB.DTQF)

Bits 16-23 — User status — defined by the destination task.

Bits 24-31 Options — contains user-request control specification as follows:

Bit	Meaning if Set
24	request is to be issued in no-wait mode (PSB.NWM)
25	do not post completion status or accept return parameters. This bit is examined only if bit 24 is set. When this bit is set, the request was issued in the no call back mode. (PSB.NCBM).

L.15 Resource Create Block (RCB)

Each H.VOMM entry point that creates a permanent file, a temporary file, a memory partition, or a directory may receive a resource create block (RCB) in order to fully define the attributes of the resource that is created. RCB formats are described in the next three tables. RCBs must be doubleword bounded.

If an RCB is not supplied by the caller, the resource is created with the default attributes described in the MPX-32 Reference Manual Volume I, Chapter 4.

Permanent and Temporary File Resource Create Block (RCB)

Word	0 7 8 15 16 23 24 31
0, 1	File owner name (RCB.OWNR)
2, 3	File project group name (RCB.USER)
4	Owner rights specifications (RCB.OWRI). See Note 1.
5	Project group rights specifications (RCB.UGRI). See Note 1.
6	Other's rights specifications (RCB.OTRI). See Note 1.
7	Resource management flags (RCB.SFLG). See Note 2.
8	Maximum extension increment (RCB.MXEI). See Note 3.
9	Minimum extension increment (RCB.MNEI). See Note 4.
10	Maximum file size (RCB.MXSZ). See Note 5.
11	Original file size (RCB.OSIZ). See Note 6.
12	File starting address (RCB.ADDR). See Note 7.
13	File RID buffer (RCB.FAST). See Note 8.
14	Option flags (RCB.OPTS). See Note 9.
15	Default override (RCB.FREE). See Note 10.

Notes:

1. Rights specifications are optional:

Bit	Description
0	read access allowed (RCB.READ)
1	write access allowed (RCB.WRIT)
2	modify access allowed (RCB.MODI)
3	update access allowed (RCB.UPDA)
4	append access allowed (RCB.APPN)
9	delete access allowed (RCB.DELE)

2. Resource management flags. For any bit not set, system defaults apply and, in some cases, the default is the equivalent of the bit being set (optional):

Bit	Description
0-7	resource type, equivalent to file type code, interpreted as two hexadecimal digits, 0 - FF (RCB.FTYP)
8-10	reserved
11	file EOF management required (RCB.EOFM)
12	fast access (RCB.FSTF)
13	do not save (RCB.NSAV)
14	reserved for MPX-32 usage
15	file start block requested (RCB.SREQ)
16	file is executable (RCB.EXEC)
17	owner ID set on access (RCB.OWID)
18	project group ID set on access (RCB.UGID)
19	reserved
20	maximum file extension increment is zero. System default value not used. (RCB.MXEF)
21	minimum file extension increment is zero. System default value not used (RCB.MNEF)
22	reserved
23	zero file on creation/extension (RCB.ZERO)
24	file automatically extendible (RCB.AUTO)
25	file manually extendible (RCB.MANU)
26	file contiguity desired (RCB.CONT)
27	shareable (RCB.SHAR) (owner rights spec only)
28	link access (RCB.LINK)
29-30	reserved
31	file data initially recorded as blocked (RCB.BLOK)

3. Maximum extension increment is the desired file extension increment specified in blocks (optional). Default is 64 blocks.
4. Minimum extension increment is the minimum acceptable file extension increment specified in blocks (optional). Default is 32 blocks.
5. Maximum file size is the maximum extendible size for a file specified in blocks (optional).
6. Original file size is the original file size specified in blocks (optional). Default is 16 blocks.

7. File starting address is the disk block where the file should start, if possible. If the space needed is currently allocated, an error is returned (optional).
8. File RID buffer is the address within the file creator's task where the eight word resource identifier (RID) is to be returned. If this parameter is not supplied (i.e., is zero), the RID for the created file is not returned to the creating task.
9. Option flags bits are as follows:

Bit	Description
0	owner has no access rights (RCB.OWNA)
1	project group has no access rights (RCB.USNA)
2	others have no access rights (RCB.OTNA)

10. Default override - If set, these bits override any corresponding bit set in RCB.SFLG and the system defaults (optional):

Bit	Description
0-7	must be zero
8-10	reserved
11	file EOF management not required
12	fast access not required
13	resource can be saved
14-22	reserved
23	do not zero file on creation/extension
24	file is not automatically extendible
25	file is not manually extendible
26	file contiguity is not desired
27	resource is not shareable
28-30	reserved
31	file data initially recorded as unblocked

Directory Resource Create Block (RCB)

	0 7 8 15 16 23 24 31
Word 0-1	Directory owner name (RCB.OWNR)
2-3	Directory project group name (RCB.USER)
4	Owner rights specifications (RCB.OWRI). See Note 1.
5	Project group rights specifications (RCB.UGRI). See Note 1.
6	Other's rights specifications (RCB.OTRI). See Note 1.
7	Resource management flags (RCB.SFLG). See Note 2.
8-10	Reserved
11	Directory original size (RCB.OSIZ). See Note 3.
12	Directory starting address (RCB.ADDR). See Note 4.
13	Directory RID buffer (RCB.FAST). See Note 5.
14	Option flags (RCB.OPTS). See Note 6.
15	Default override (RCB.FREE). See Note 7.

Notes:

1. Rights specifications bits are as follows:

Bit	Description
0	read access allowed (RCB.READ)
8	directory may be traversed (RCB.TRAV)
9	directory may be deleted (RCB.DELE)
10	directory entries may be deleted (RCB.DEEN)
11	directory entries may be added (RCB.ADEN)

2. Resource management flags are optional:

Bit	Description
13	do not save (RCB.NSAV)
27	shareable (RCB.SHAR)

3. Directory original size is the number of entries required (optional).
4. Directory starting address is the disk block number where the directory should start, if possible. If the space needed is currently allocated, an error is returned (optional).
5. Directory RID buffer is the address within the directory creator's task where the eight word resource identifier (RID) is to be returned. If this parameter is not supplied (i.e., is zero), the RID for the created directory is not returned to the creating task.

6. Option flags are as follows:

Bit	Description
0	owner has no access rights (RCB.OWNA)
1	project group has no access rights (RCB.USNA)
2	others have no access rights (RCB.OTNA)

7. If default override is set, these bits override any corresponding bit set in RCB.SFLG and the system defaults (optional).

Bit	Description
0-7	must be zero
13	resource can be saved
27	resource is not shareable

Memory Partition Resource Create Block (RCB)

Word	0 7 8 15 16 23 24 31
Word 0-1	Partition owner name (RCB.OWNR)
2-3	Partition project group name (RCB.USER)
4	Owner rights specifications (RCB.OWRI). See Note 1.
5	Project group rights specifications (RCB.UGRI). See Note 1.
6	Other's rights specifications (RCB.OTRI). See Note 1.
7	Resource management flags (RCB.SFLG). See Note 2.
8-9	Reserved
10	Starting word page number (RCB.PPAG)
11	Partition original size (RCB.OSIZ). See Note 3.
12	Partition starting address (RCB.ADDR). See Note 4.
13	Partition RID buffer (RCB.FAST). See Note 5.
14	Option flags (RCB.OPTS). See Note 6.
15	Default override (RCB.FREE). See Note 7.

Notes:

1. Rights specifications are optional:

Bit	Description
0	read access allowed (RCB.READ)
1	write access allowed (RCB.WRIT)
9	delete access allowed (RCB.DELE)

2. Resource management flags are optional:

Bit	Description
13	do not save (RCB.NSAV)

3. Partition's original size is the number of protection granules required.
4. Partition's starting address is a 512-word protection granule number in the user's logical address space where the partition is to begin.
5. Partition's RID buffer is the address within the partition creator's task where the eight word resource identifier (RID) is to be returned. If this parameter is not supplied (i.e., is zero), the RID for the created partition is not returned to the creating task.
6. Option flags are optional:

Bits	Description
0	owner has no access rights (RCB.OWNA)
1	project group has no access rights (RCB.USNA)
2	others have no access rights (RCB.OTNA)
9	defines a static partition (RCB.STAT)
24-31	define memory class (RCB.MCLA). Values are:

Value	Memory Class
0	S (default)
1	E
2	H
3	S

7. If set, these bits override any corresponding bit set in RCB.SFLG and the system defaults (optional):

Bits	Description
0-7	must be zero
13	resource can be saved

L.16 Resource Identifiers (RID)

The fastest means of locating a volume resource (once created) is by its resource identifier (must be on a doubleword boundary). The resource identifier has the following format:

Word	0 7	8 15	16 23	24 31
Word 0-3	Volume name			
4	Creation date			
5	Creation time			
6	Volume address of resource descriptor			
7	Must contain zero		Resource type	

Since the resource identifier contains the volume address of the resource descriptor, the resource descriptor (which points to and describes the resource) can be accessed directly without going through the various directories which would otherwise have to be traversed.

Given a valid pathname defining a resource, the corresponding resource descriptor may be retrieved by the H.VOMM locate resource service. The first eight words of a resource descriptor consist of the resource identifier.

L.17 Resource Logging Block (RLB)

The resource logging block (RLB) is a word-bounded data structure used to pass information between H.VOMM and the caller. The information is used to locate a directory entry and resource descriptor for a single resource or for all resources defined in a particular directory.

	0 ... 7	8 ... 15	16 ... 23	24 ... 31
Word 0	Pathname vector or RID address (RLB.TGT)			
1	Resource directory buffer address (192W) (RLB.BUFA). See Note 1.			
2	Associated mounted volume table entry address (RLB.MVTE)			
3	Parent directory RD block address (RLB.RDAD)			
4	Type (RLB.TYPE). See Note 2.	Buffer offset (RLB.BOFF)		
5	Length. See Note 3.	Return buffer address (RLB.DIRA)		
6	User FCB address (RLB.FCB)			
7	Flags. See Note 4.	Reserved (RLB.INT)		

Notes:

1. Optional. If not specified, a resource directory is not returned.
2. Bits in RLB.TYPE are assigned as follows:

Bits	Meaning if Set
0	indicates recall (RLB.RECA)
1-7	reserved

3. This word contains the address of a buffer and its length in words (the buffer can be up to 16 words long).
4. Bits in the flags byte are assigned as follows:

Bits	Meaning if Set
0-1	reserved
2	directory entry and resource descriptor for specified directory are returned
3	root directory
4	resource is located
5-7	reserved

L.18 Resource Requirement Summary (RRS) Entries

The resource requirement summary (RRS) is a doubleword bounded data structure used to identify the resources required by a task to the resource manager. Resources are statically allocated using the information in the RRS entry. The RRS is generally built by processors requiring static allocation of resources, such as TSM, cataloger, etc., or supplied as an argument for dynamic allocation.

For compatibility purposes, revision 1.*x* RRS formats can be used. The details of these formats can be found in Chapter 2 of a revision 1.*x* Technical Manual.

Type 1 - Assign by Pathname

	0 7	8 15	16 23	24 31
Word 0	Zero	Logical file code (RR.LFC)		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Plength (RR.PLEN)	Reserved. See Note 2.
2	Access (RR.ACCS). See Note 3.			
3	Options (RR.OPTS). See Note 4.			
4-*n*	Pathname (variable length) (RR.NAME1)			

Type 2 - Assign to Temporary File

	0 7	8 15	16 23 24 31
Word 0	Zero	Logical file code (RR.LFC)	
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Initial file size (RR.PLEN)
2	Access (RR.ACCS). See Note 3.		
3	Options (RR.OPTS). See Note 4.		
4-7	Volume name (16 characters; left-justified, blank-filled) (RR.NAME1) (Volume name is optional)		

Type 3 - Assign to Device

	0 – 7	8 – 15	16 – 23	24 – 31
Word 0	Zero	Logical file code (RR.LFC)		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Density (RR.DENS). See Note 5.	Zero
2	Access (RR.ACCS). See Note 3.			
3	Options (RR.OPTS). See Note 4.			
4	Device type (RR.DT3). See Note 6.	Volume number (RR.VLNUM)	Channel number See Note 7. (RR.CHN3)	Subchannel number (RR.SCHN3)
5	Unformatted ID (1-4 characters) (RR.UNFID)			

Type 4 - Assign to LFC

	0 – 7	8 – 15	16 – 23	24 – 31
Word 0	Zero	Logical file code (RR.LFC)		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Zero	
2	Zero	Logical file code (RR.SFC)		
3	Options (RR.OPTS). See Note 4.			

Type 5 - Assign by Segment Definition

	0 – 7	8 – 15	16 – 23	24 – 31
Word 0	Zero	Logical file code (RR.LFC)		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	UDT index (RR.UDTI)	Reserved
2	Access (RR.ACCS). See Note 3.			
3	Options (RR.OPTS). See Note 4.			
4	Starting block number (RR.STBLK)			
5	Number of blocks (RR.NBLKS)			

Type 6 - Assign by Resource ID

Word	0–7	8–15	16–23	24–31
Word 0	Zero	Logical file code (RR.LFC)		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Zero	Reserved
2	Access (RR.ACCS). See Note 3.			
3	Options (RR.OPTS). See Note 4.			
4-7	Volume name (16 characters; left-justified, blank-filled) (RR.NAME1)			
8	Binary creation date (RR.DATE)			
9	Binary creation time (RR.TIME)			
10	Resource descriptor block address (RR.DOFF)			
11	Reserved		Resource type (RR.RTYPE)	

Type 7 - Reserved for Future Use

Type 8 - Reserved for Future Use

Type 9 - Mount by Device Mnemonic

Word	0–7	8–15	16–23	24–31
Word 0	Zero	System ID (RR.SYSID). See Note 11.		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Zero	
2	Access (RR.ACCS). See Note 3.			
3	Options (RR.OPTS). See Note 4.			
4-7	Volume name (16 characters; left-justified, blank-filled) (RR.NAME1)			
8	Device type (RR.DT9). See Note 8.	Reserved	Channel number (RR.CHN9). See Note 9.	Subchannel number (RR.SCHN9)
9	Zero			

Type 10 - Assign to ANSI Tape

	0 7	8 15	16 23	24 31
Word 0	Zero	Logical file code (RR.LFC)		
1	Type (RR.TYPE). See Note 1.	Size (RR.SIZE)	Format (RR.FORM)	Protect (RR.PROT)
2	Access (RR.ACCS). See Note 3.			
3	Options (RR.OPTS). See Note 4.			
4	Record length (RR.RECL)		Block size (RR.BSIZE)	
5	Generation number (RR.GENN)			
6	Generation version number (RR.GENV)			
7	Absolute termination date (RR.EXPIA)			
8	Relative termination date (RR.EXPIR)		Logical volume identifier (RR.LVID)	
9	RR.LVID (cont.)			
10-13	17-character file identifier (RR.AFID)			
14	RR.AFID (cont.)	Reserved		
15	Reserved			

Type 11 - Assign to Shadow Memory

	0 7	8 15	16 23	24 31
Word 0	Zero			
1	Type (RR.TYPE) See Note 1.	Size (RR.SIZE)	Shadow flags (RR.SHAD). See Note 10.	
2	Start address (RR.SADD)			
3	End address (RR.EADD)			

Notes:

1. Bits in RR.TYPE are assigned as follows:

Value	Meaning
1	assign by pathname (RR.PATH)
2	assign to temporary file (RR.TEMP)
3	assign to device (RR.DEVC)
4	assign to secondary LFC (RR.LFC2)
5	assign to segment definition (RR.SPACE)
6	assign by resource ID (RR.RID)
7	reserved for future use
8	reserved for future use
9	mount by device mnemonic (RR.MTDEV)
10	assign to ANSI labeled tape (RR.ANS)
11	assign to shadow memory (RR.SHTYP)
12-255	reserved

2. Byte 3 is zero. This field is used by MPX-32 for big blocking buffers.
3. Bits in RR.ACCS are assigned as follows:

Bits	Meaning if Set
0	read access allowed (RR.READ)
1	write access allowed (RR.WRITE)
2	modify access allowed (RR.MODFY)(not valid for ANSI tapes)
3	update access allowed (RR.UPDAT)
4	append access allowed (RR.APPND)
5-15	reserved
16	explicit shared use requested (RR.SHAR)
17	exclusive use requested (RR.EXCL)
18	assign as volume mount device (RR.MNT)
19-31	reserved

4. Bits in RR.OPTS are assigned as follows:

Bits	Meaning if Set
0	treat as SYC file (RR.SYC) (TSM/JOB only)
1	treat as SGO file (RR.SGO) (TSM/JOB only)
2	treat as SLO file (RR.SLO)
3	treat as SBO file (RR.SBO)
4	explicit blocked option (RR.BLK)
5	explicit unblocked option (RR.UNBLK)
6	inhibit mount message (RR.NOMSG)
7	reserved for system use
8	automatic open requested (RR.OPEN)
9	user-supplied blocking buffer address in FCB (RR.BUFF)
10-11	reserved for system use
12	mount with no-wait (RR.NOWT)
13	mount as public volume (RR.PUBLC)
14	set by H.VOMM for special case handling of VOMM assignments (RR.VOMM)
15	file is spooled when deallocated (RR.SEP)
16	ANSI labeled tape on RRS type 3 (RR.ANSI)
17-31	reserved

5. RR.DENS contains the density specification for XIO high speed tape units. When specified, this field has the following bit significance:

Bits	Meaning if Set
0	indicates 800 bpi nonreturn to zero inverted (NRZI)
1	indicates 1600 bpi phase encoded (PE)
6	indicates 6250 bpi group coded recording (GCR)

If this field is zero, 6250 BPI is set by default.

6. RR.DT3 specifies whether or not a channel is present and specifies the device type:

Bits	Meaning if Set
0	channel present
1-7	device type

7. RR.CHN3 specifies whether or not a subchannel is present and specifies the channel number:

Bits	Meaning if Set
0	subchannel is present. Examined only if bit zero of RR.DT3 is set.
1-7	channel number

8. RR.DT9 specifies whether or not a channel is present and specifies the device type:

Bits	Meaning if Set
0	channel present
1-7	device type

9. RR.CHN9 specifies whether or not a subchannel is present and specifies the channel number:

Bits	Meaning if Set
0	subchannel is present. Examined only if RR.DT9 is set.
1-7	channel number

10. RR.SHAD contains the shadow flags that qualify the start and end addresses, or specify what portions of the task are to be shadowed:

Bits	Meaning if Set
0-7	reserved
8	shadow the task (RR.SHTSK)
9	shadow the TSA (RR.SHTSA)
10	shadow the stack (RR.SHST)
11	shadow memory is required (RR.SHRQ)
12	shadow the entire task (RR.SHALL)
13	absolute address (RR.ABS)
14	relative to the code section origin (RR.CREL)
15	relative to the data section origin (RR.DREL)

11. RR.SYSID is the ID for mounting a multiprocessor volume. Valid IDs are:

Multiported (MP)	0 through F
Dual Ported (DP)	0 or 1

For more information on mounting multiprocessor volumes see the MPX-32 Reference Manual Volume I, Chapter 4, Mounting Multiprocessor Volumes.

L.19 Receiver Exit Block (RXB)

The receiver exit block (RXB) is used to control the return of parameters and status from the destination (receiving) task to the task that issued the send request. It is also used to specify receiver exit options. The same format RXB is used for both messages and run requests. The address of the RXB must be presented as an argument when either the M.XMSGR or M.XRUNR services are called.

	0 — 7	8 — 15	16 — 23	24 — 31
Word 0	Return status (RXB.ST)	Return parameter buffer address (RXB.RBA)		
1	Options (RXB.OPT)	Reserved	Number of bytes to be returned (RXB.RQ)	

Notes:

1. Return status (RXB.ST) contains status as defined by the receiver task. Used to set the user status byte in the parameter send block (PSB) of the task which issued the send request.
2. Return parameter buffer address (RXB.RBA) contains the word address of the buffer containing the parameters which are to be returned to the task which issued the send request.
3. Options (RXB.OPT) contains receiver exit control options. It is encoded as follows:

Value	Exit Type	Meaning
0	M.XRUNR	wait for next run request.
	M.XMSGR	return to point of task interrupt.
1	M.XRUNR	exit task, process any additional run requests. If none exist, perform a standard exit.
	M.XMSGR	N/A

4. Number of bytes to be returned (RXB.PQ) contains the number of bytes (0 to 768) of information to be returned to the sending task.

L.20 Type Control Parameter Block (TCPB)

The type control parameter block (TCPB) allows I/O to and from the system console by setting up task buffer areas for messages output by a task and optional reads back from the console. If no input is desired, word one of the TCPB must be zero.

See the MPX-32 Reference Manual Volume I, Chapter 5 for further details on the TCPB.

	0 — 11	12	13 — 31
Word 0	Output quantity (TCP.OQ)	See Note 1.	Output data address (TCP.OTCW)
1	Input quantity (TCP.IQ)	See Note 1.	Input data address (TCP.ITCW)
2	Console Teletype Flags (TCP.FLGS). See Note 2.		

Notes:

1. Bit 12 is set to 1.
2. Bits in TCP.FLGS are assigned as follows:

Bits	Meaning if Set
0	no-wait I/O
31	operation in progress. This bit is reset after post-I/O processing completes.

Type Control Parameter Block (TCPB) using 24-bit address:

Word	0–7	8–15, 16–23, 24–31
Word 0	Output quantity (TCP.OQ)	Output data buffer address (TCP.OTCW)
1	Input quantity (TCP.IQ)	Input data buffer address (TCP.ITCW)
2	Console device flags (TCP.FLGS) See Note 1.	

Notes:

1. Bit interpretations for TCP.FLGS are:

Bits	Meaning if Set
0	no-wait I/O
1	data buffer addresses are 24-bit addresses (TCP.LAD) Note: This bit must be set.
31	operation in progress. This bit is reset after post-I/O processing completes.

L.21 Unit Definition Table (UDT)

The unit definition table (UDT) is a system resident structure that identifies device-dependent information required by a handler for a specific device. The UDT is built by the SYSGEN process, one for each device configured in the system. During SYSGEN, each UDT is linked to its corresponding controller definition table (CDT) and its associated controller and handler.

Word	0 – 7	8 – 15	16 – 23	24 – 31
0	UDT index (UDT.UDTI)		CDT index (UDT.CDTI)	
1	Unit status (UDT.STAT). See Note 1.	Device type code (UDT.DTC). See Note 2.	Logical channel number (UDT.CHAN)	Logical subaddress (UDT.SUBA)
2	Reserved	Address of dispatch queue entry of task which has device allocated if device is not shared (UDT.DQEA)		
3	Physical channel number (UDT.PCHN)	Physical subaddress (UDT.PSUB)	Sectors per block (UDT.SPB) or number of characters per line (UDT.CHAR). See Note 3.	Sectors per allocation unit (UDT.SPAU) or number of lines per screen (UDT.LINE). See Note 4.
4	Flags (UDT.FLGS). See Note 5.	Number of sectors per track on disk or global line counter if a terminal (UDT.SPT)	Maximum byte transfer (UDT.MBX)	
5	Number of sectors on disk or tab setting if a terminal (UDT.SECS)			
6	Sector size, on disk or a tab setting if a terminal (UDT.SSIZ)		Number of heads on disk or a tab setting if a terminal (UDT.NHDS)	
7	Serial number if tape or removable disk (UDT.SERN). See Note 6.			
8	Peripheral time-out value (UDT.PTOV)			
9	Reserved	Address of device context area (UDT.DCAA) or handler name at initialization (UDT.HNAM)		
10	Bit flags (UDT.BIT2). See Note 7.		Associated allocated resource table index if assigned (UDT.ARTI)	
11	Service interrupt handler address (UDT.SIHA)			
12	Reserved (UDT.CXR) See Note 8.	Secondary flags (UDT.BIT3) See Note 9.	Reserved (UDT.SHFL)	Reserved (UDT.DQEN)
		or UDT.HIST. See Note 10		
13	Address of first IOQ linked to this device (UDT.FIOQ)			
14	Address of last IOQ linked to this device (UDT.BIOQ)			
15	Link Priority (UDT.LPR1)	Link Count (UDT.IOCT)	Unit Status byte 2 (UDT.STA2). See Note 11.	

Notes:

1. Bits in UDT.STAT are assigned as follows:

Bit	Meaning if Set
0	online (UDT.ONLI)
1	dual-portd XIO disk (UDT.DPDC)
2	allocated (UDT.ALOC)
3	terminal in use and not in wait (UDT.USE)
4	system output unable to allocate (UDT.NOAL)
5	shared device (UDT.SHR)
6	premounted (UDT.PREM)
7	terminal (TSM) device (UDT.TSM)

2. For example, 01 for any disk, 04 for any tape, etc. Valid device type codes are listed in Appendix A.
3. For disks, contains the number of sectors per block (UDT.SPB). For terminals, contains the number of characters per line (UDT.CHAR).
4. For disks, contains the number of sectors per allocation unit (UDT.SPAU). For SLO or terminals, contains the number of lines per page or screen (UDT.LINE).
5. Bits in UDT.FLGS are assigned as follows:

Bit	Meaning if Set
0	extended I/O device (UDT.FCLS)
1	I/O outstanding (UDT.IOUT)
2	removable disk pack (UDT.RMDV)
3	a break has been requested for this device (UDT.LOGO)
4	autoselectable for batch SLO (UDT.BSLO)
5	autoselectable for batch SBO (UDT.BSBO)
6	autoselectable for real-time SLO (UDT.RSLO)
7	autoselectable for real-time SBO (UDT.RSBO)

6. If the device is a terminal or console, the first halfword is the current terminal type for TERMDEF (UDT.CTDF) and the second halfword is the default terminal type (UDT.DTDF).
7. Bits in UDT.BIT2 are assigned as follows:

Bits	Meaning if Set
0	port is private; else switched (UDT.DIAL)
1	port is connected to modem (UDT.MODM)
2	port has graphic capability (UDT.GRFC)
3	port is full duplex (UDT.FDUX)
4	port is configured multidrop (UDT.MDRA)
5	volume mounted on device (UDT.VOL)
6	echo by computer (UDT.ECHO)
7	device has failed. Log off TSM (UDT.DEAD)
8	cache device (UDT.CAC)
9	inhibit automatic line wrap (UDT.NRAP)
10	spool device requires form feed after printing rather than before; initial form feed is inhibited (UDT.FEOP)

Bits	Meaning if Set
11	quarter inch cartridge tape drive (UDT.QITD)
12	software read flow control required (UDT.RXON)
13	software write flow control required (UDT.WXON)
14	hardware read flow control required (UDT.RHWF)
15	hardware write flow control required (UDT.WHWF)

8. For switched port, contains the value specified in the LOGONFLE CXR = option (UDT.CXR)
9. Bits in UDT.BIT3 are assigned as follows:

Bits	Meaning if Set
0	SCSI device (UDT.SCSI)
1-7	reserved

10. UDT.HIST is used as an address save area by pseudo device handlers, such as ON.IPXIO
11. Bits in UDT.STA2 are assigned as follows:

Bits	Meaning if Set
0	IOQ linked from UDT (UDT.IOQ)
1	IOP device (initialized by SYSGEN) (UDT.IOP)
2	device malfunction (UDT.MALF)
3	operator intervention applicable (UDT.INTV)
4	use standard XIO interface
5	floppy disk
6	cartridge module drive
7	moving head disk with fixed head option
8	if software read flow control enabled, use DTR line; otherwise, use RTS line. (UDT.RDTR)
9	memory disk (UDT.MD) or valid command line recall and edit device (UDT.CLRE)
10	memory allocated for memory disk (UDT.MDAL)
11	start address of memory disk specified at SYSGEN (UDT.MDST)
12	multiport device is shared with an MPX-32 Revision 3.2C or earlier version (UDT.PPV)
13	device is exclusive ANSI (UDT.ANSI)
14	serial printer (UDT.SLPR)
15	port is switched and CXR=N option has been specified (UDT.DCXR)

Glossary

access method	A software package that provides the ability to access fields within records, to classify or order records according to the contents of fields, and to perform other such functions.
access mode	Defines the range of operations to be performed on a resource.
aged page	A page which has not been referenced within a predetermined frame of time during demand page processing. This page is no longer considered a part of the task's working set.
allocated resource table (ART)	A system resident table with an entry for each currently allocated resource in the system.
allocation	The process of securing a resource for a specific usage and access mode for a task.
allocation unit	A mechanism for grouping more than one block on a formatted disc, or other mass medium, at one time. Usually specified in multiples of 192-word disc blocks. See disc block.
argument	A value (string or integer) that is assigned to a parameter.
assign	To associate a resource with a logical file code used by a process.
assignment	The process of associating a logical file code with a system resource. Does not guarantee the resource for a specific use or access mode for a task.
asynchronous	Implies one entity does not wait for or otherwise acknowledge another entity when it performs an operation.
asynchronous notification	A process does not stop execution waiting for notification. It receives a software interrupt when an asynchronous operation is complete.
base mode	Implies the base register instruction set that allows executable programs of up to 4096KW (16MB).
blocked I/O	The process of packing records equal to or less than 254 bytes so that more than one record is stored in a 192-word disc block.
blocking buffers	Buffers used for packing records for blocked I/O. See blocked I/O.

caller notification packet (CNP)	A structure used to supply additional calling parameters and to control the handling of abnormal conditions that may occur during resource requests.
classes of users	A three-level grouping of users into OWNER, PROJECTGROUP and OTHER. Used to permit or limit access to a resource by 'class'.
command file	A file containing commands known to a particular operating system or process.
CONCEPT/32	A term which implies the entire line of CONCEPT/32 computers; for example, the 32/67.
configuration	Hardware: the physical hardware related to a CPU. Software: adapting the operating system to a hardware configuration with the SYSGEN processor.
data files	Files containing data or transactions that have been processed or will be processed by a task.
data management	The ability to structure data into records using buffers.
Datapool	An area of memory that contains the same functionality as Global Common but with the added flexibility of symbolic references being independent of the actual positioning of data within the memory area. See Global Common.
deallocate	To detach a resource from a process.
deassign	To remove the association between a logical file code and a resource and deallocate the resource.
dequeue	To remove from a prioritized list.
demand page	Allocation of memory when the logical page is referenced by the task on demand. The process of allocating physical memory when pages are referenced and deallocating physical memory when pages are no longer active. Pages that are no longer active are considered aged and removed from the task's working set.
device	A peripheral unit such as a card reader, a printer, a disc drive, or a tape drive. Distinguished from media used with devices.
device access	Levels are physical I/O, logical device I/O, and logical file I/O.
device-dependent I/O	Tasks perform operations to a specified device with minimal IOCS overhead.
device-independent I/O	Tasks perform I/O operations through the use of operating system calls which are independent of the device used to perform the operation.

direct I/O	Tasks perform operations bypassing IOCS and handler functions by coding its own handler and attaching it to a specific channel.
directory	A list of file names and/or memory partition names. Stored on disc like a regular file. Located via a resource descriptor for the directory. Directory names are 1 to 16 characters in length and valid characters for names are A to Z, 0 to 9, dot (.) and underscore (_).
directory descriptor	The resource descriptor for a directory.
disc block	A common unit of measurement (some number of words) used to measure file space on formatted media throughout a system. The number of words in a block is oriented to the most common sector size on discs used with the system.
DMAP	See resource descriptor allocation map.
dynamic assignment	The association of a logical file code with a system resource during task execution.
enqueue	To put into a list ordered by software priority.
exclusive use	A resource is not available for use by any other task until that resource is deallocated by the using task. Guarantees access to a resource, within the access limitations imposed by the resource creator, when logical I/O is initiated.
executable image	A file of object code produced by the LINKER/X32.
explicit shared use	A resource can be used concurrently by more than one task. Each task maintains resource integrity by establishing its own synchronization and locking mechanisms. Each task is guaranteed access to the resource, within access limitations imposed by the resource creator, when logical I/O is initiated.
extended code	That part of the operating system that has been modified to run in the extended execution space.
extended file control block	A file control block set up by the user which contains more information than the standard file control block. See file control block.
file	A set of information stored on a mass medium such as disc or tape that is given a unique identity (number and often name) and treated as a single entity for processing.
file control block (FCB)	Set up by the user to describe logical files within a task. Describes attributes of logical I/O operation.
file descriptor	A resource descriptor for a file.

file identifier	A unique identifier stored in the resource descriptor for a file when the file is created. Used to access the resource descriptor without a directory search.
file segment	Set of contiguous allocation units on a volume identifying the space associated with a file. Each file segment definition contains the absolute 192-word block volume segment address and the segment length in 192-word blocks (maximum of 32 file segment definitions per file).
file space allocation map (SMAP)	A bit map used to allocate space on a volume.
filename	A 1- to 16-character name supplied for a permanent file when it is created on a mass medium. Used in most cases thereafter to identify the file. Valid characters for filenames are the upper-case letters A to Z, the decimal numbers 0 to 9, and the special characters dot (.) and underscore (_). Filenames to be used with the compatible interfaces, for example Editor, File Manager, and Media, are limited to 1 to 8 characters.
format	Standard organization of information.
formatted volume	A disc pack or floppy disc that contains standard volume system structures established by the Volume Formatter utility.
Global Common	An area of memory accessible by using symbolic names to identify specific storage cells. Programs belonging to many independent tasks can freely access the same data and exchange control information within the Global Common area.
implicit shared use	A resource is available for concurrent use by other tasks in a compatible access mode. Does not guarantee access when logical I/O is initiated. Resource integrity is automatically maintained by the system.
job file	A command file designed to run in the batch or interactive environment.
library file	Object modules or source modules identified by name that are output to a single file. Modules on library files can be used separately and repeatedly. For example, object modules can be retrieved by name during cataloging and inserted with existing code. The ability to edit the contents of library files by name is also normally available.
load module file	A file of object code produced by the Cataloger that is ready to relocate from disc into memory and execute as a process. Load module files can be activated by name and are controlled by name or task number.

logical device I/O	I/O where the physical characteristics of a device are not determined automatically by the file management system (device and data formatting are inhibited), allowing the user to exert control over a particular physical device or device medium.
logical dismount	The action taken by MPX-32 to disassociate a volume from the requesting task. A TSM logical dismount disassociates the volume from the requesting context.
logical file code (LFC)	User defined 1- to 3-character ASCII codes identifying logical files within tasks.
logical file I/O	I/O where the physical characteristics of a device and device medium (device format control, data conversion, data formatting) are performed automatically for the user so that he gains a degree of device independence.
logical mount	The action taken by MPX-32 to associate a physically mounted volume to a task. A TSM logical mount associates the volume to the TSM context requesting the mount.
logical resource	Any entity existing only because of a mechanism provided by software. The primary logical resources are: disc volumes, directories, files, and memory partitions.
map block	A 2048-word unit of memory allocation. In demand page processing, a page is a map block.
medium (singular) media (plural	A contiguous source of input or output that is used for a particular peripheral device. For example, a disc pack is the medium mounted on a disc drive; a tape is the medium mounted on a tape drive; paper is the medium used on a printer; a deck of cards is the medium used on a card reader. The operating system distinguishes use of media from use of devices.
memory descriptor	The resource descriptor for a memory partition.
memory partitions	Named areas of physical memory that can be shared by concurrently executing tasks.
modular	Construction in independent layers. Each higher level layer builds on the layer beneath it and provides its own standard interfaces to the levels above and below it.
mounted volume table (MVT)	A system resident table with an entry for each physically mounted volume. Each entry contains information used by the system to maintain volume accounting information.

multicopied tasks	Tasks with the same name and the same concurrent load module activity, owned by a single owner or several owners. This is accomplished by cataloging a task as multicopy. Task numbers must be used to communicate with multicopied tasks. See task number.
multiprocessor volume	A specially mounted user volume that allows tasks operating in separate system environments to concurrently access any volume resource.
multivolume magnetic tape	A set of 1 through 255 maximum physical reels of magnetic tape processed as a continuous reel.
nonbase mode	Implies the nonbase register instruction set which allows executable programs of up to 128KW.
nonpublic volume	A volume assigned specifically to the tasks that mount it. Remains physically mounted until use and assign counts equal 0.
object file	A file of assembled or compiled code that can be cataloged or linked into a task.
owner	The user who has possession of and can control access to a file, device, memory partition, or directory. Usually the owner of a resource is the user who created its resource descriptor.
owner name	A 1- to 8-character name supplied at logon which remains unchangeable through logoff. The following characters cannot be used in owner names: blanks, commas, semicolons, equal signs, line feeds, dollar signs, percent signs, exclamation points, and left or right parentheses. All other characters are valid. Owner names are associated with any task or process activated on the system and noted by any process that acts in the owner's behalf. Owner name is also associated with any resources a user creates unless the user specifies otherwise. Specifying a different owner when creating a resource definition does not change the user's owner name; it only specifies the owner name associated with the resource.
page	A 512-word unit of memory protection. Also referred to as a protection granule. Four pages compose a map block. For demand page processing, a page is a map block brought into memory and removed from memory during the life of a demand page task.
page fault	The reference of a page within the logical address space which is not currently a part of the task's working set.

page in	Bringing into logical memory a page needed to satisfy an address referenced by a task.
page out	The removal of aged pages from the task's logical address space.
parameter	A symbolic name in a process or directive file that can be assigned an argument.
pathname	Variable length ASCII character strings which uniquely identify a volume resident resource by explicitly or implicitly describing the volume, one or more directories, and the resource name.
pathname block	Doubleword bounded variable length ASCII character string beginning with "!" which uniquely identifies a volume resident resource by explicitly or implicitly describing the volume, one or more directories, and the resource name.
permanent files	Files that remain defined on a volume until explicitly deleted.
physical dismount	The action taken by MPX-32 to disassociate a volume from an assigned mount device and deallocate the device.
physical mount	The action taken by MPX-32 to allocate a mount device and associate that device to the assigned volume name.
physical resource	Any physical hardware that MPX-32 supports. Tasks access the resource to perform their functions. The primary physical resources are: the CPU, computer memory (main storage), and input/output devices.
portable	Can be used on any compatible device in a single system configuration. Can also be carried to a compatible device on a different system hardware configuration. Usually describes a volume.
post program-controlled interrupt receiver	User supplied end-action receiver entered when a hardware post program-controlled interrupt is encountered.
process	A body of code scheduled for CPU time as a single entity. A load module is a process, in loadable form, stored on disc. Same as task.
project group name	A name that is specified at logon and can also be changed. Identifies a group of users that have a defined set of rights when they access a resource.
protect	To limit access to a resource. See classes of users.
protection granule	A 512-word unit of memory protection. Also referred to as a page in a non-demand page context. Four protection granules compose a map block.

public volume	A volume available for resource assignments by all tasks activated in the system.
real time task	Synonymous with time critical process.
requestor	The process which requests a function. Each process on a system has an associated owner name. The system process that requests a function for a user (e.g., in the interactive environment) keeps track of the owner name so that the user thinks of himself as the 'requestor'.
resource	Any source of support that exists external to a task and that the task needs to perform its function. A resource can be physical or logical.
resource create block (RCB)	Defines access attributes for permanent files, temporary files, memory partitions, and directories when the particular resource is created. If not supplied at resource creation, system default attributes are assumed.
resource descriptor (RD)	Contains access, accounting, and space definition information pertaining to mounted volume resources, permanent files, temporary files, directories and partitions.
resource descriptor allocation map (DMAP)	A bit map used for the allocation of resource descriptors on a volume.
resource identifier (RID)	The fastest way to locate an already created volume resource. The RID is in the first eight words of a resource descriptor and contains the volume address of the resource descriptor, which points to and describes the resource.
resource logging block (RLB)	A parameter block used as input to the M.LOGR service for logging resources.
Resource Management Module (H.REMM)	Performs allocation and assignment of all system resources and maintains access compatibility and usage rights for these resources. Also contains synchronization mechanisms for concurrent access to shared resources.
resource requirement summary (RRS)	Defines assignment requirements of a resource. Entries are variable length, doubleword bounded. There are 9 types of entries.
root directory	The directory of all directories defined on a volume.
SMAP	See file space allocation map.

source file	A file of source code to be assembled or compiled into object code.
static assignment	The association of a logical file code with a system resource during task activation.
status posting	The process of returning information that indicates whether a service was completed successfully, with errors, or denied.
swap volume	A volume used as the primary medium for swap file allocations.
symbolic	A representation of a physical resource, e.g., a name that represents an entity but is not the entity itself.
synchronous notification	A process waits on further processing until it is notified that an operation is done or that there is something inhibiting the operation (e.g., a resource is not available or other processes are in contention for the resource).
system administrator attribute (SA)	Gives an unprivileged user the ability to execute privileged SVC's, allows a user to mount public volumes, and allows a user to change his owner name. A user with the system administrator attribute is, however, restricted to resource access limitations imposed by the resource creator.
system directory	Special directory on the system volume which contains volume resources necessary for system operation.
system volume	A volume containing the system and bootstrap images from which the current system was IPLed. This volume is automatically mounted by the SYSINIT task at system initialization.
task	Synonymous with process.
task name	The name supplied when a task is cataloged or linked.
task number	An 8-digit hexadecimal number assigned to a task by MPX-32 when the task is activated. The task number is unique and identifies a particular copy or sharer of a task.
temporary files	Unnamed files that are referenced by resource identifiers. They are automatically deleted from the system and their volume space made available when the last task assigned to them terminates execution.
time critical process	A process which has time constraints. Same as a real time task.
traverse	To pass through a directory on the way to another directory or resource.

type control parameter block (TCPB)
: Set up by the user for sending and receiving messages to/from the system console.

unformatted media
: A medium (magnetic tape, disc pack or floppy disc) that does not contain valid volume format information, but must be mounted before initiation of I/O operations.

usage mode
: Defines the degree to which multiple tasks can concurrently allocate a resource. Usage modes are: exclusive use, explicit shared, and implicit shared.

user
: A person who uses a system. Processes and commands that activate processes are either initiated by a user or initiated on behalf of a user.

volume
: A medium that has a standard format. Disc packs can be formatted as volumes.

volume assignment table (VAT)
: A task resident table with an entry for each non-public volume currently assigned to the task.

Volume Management Module (H.VOMM)
: Manipulates volume resident and related memory resident structures in order to allow for creation, deletion, and maintenance of user and system resources which reside on volumes; for example, provides space management for all currently mounted volumes in the system.

working set
: The pages (map blocks) of a task that are actively being referenced within a predetermined frame of time.

Index

- B -

- C -

- D -

- G -

- H -

- I -

- J -

- N -

- O -

- P -

- R -

- S -

- U -

- W -

MPX-32™ Installation & System Administration
Reference Manual
No. 323-001553-600

READER'S COMMENT FORM

Please use this form to communicate your views about this manual. The form is preaddressed and stamped for your convenience.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy	____	____	____	____
Clarity	____	____	____	____
Completeness	____	____	____	____
Examples	____	____	____	____
Figures	____	____	____	____
Index	____	____	____	____
Organization	____	____	____	____
Retrievability of Information	____	____	____	____

Additional comments:

____

____

____

____

____

____

If you wish a reply, please print your name and mailing address:

____

____

____

____

What is your occupation/title? ______________________________

____

Thank you for your cooperation.

Note: Copies of Encore publications are available through your Encore representative or the customer service office serving your locality.

FOLD HERE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 2356 FORT LAUDERDALE, FL

POSTAGE WILL BE PAID BY ADDRESEE

ENCORE COMPUTER CORPORATION
ATTENTION: DOCUMENTATION COORDINATOR
6901 W. SUNRISE BLVD.
P.O. BOX 409148
FT. LAUDERDALE, FL 33340-9970

FOLD HERE

PLEASE TAPE DO NOT STAPLE

CUT ALONG LINE