3155-4-638-2741-3
(EXTRA COPY)

MPX-32

Release 1.5B

Reference Manual

Volume III

September 1982

Publication Order Number: 323-003663-000

GOULD
Electronics & Electrical Products

JUN83

This manual is supplied without representation or warranty of any kind. Gould Inc., S.E.L. Computer Systems Division 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.

LIMITED RIGHTS LEGEND

for

PROPRIETARY INFORMATION

The information contained herein is proprietary to Gould S.E.L. and/or its vendors, and its use, disclosure or duplication is subject to the restrictions stated in the Gould S.E.L. license agreement Form No. 620-06(1/82) or the appropriate third-party sublicense agreement. The information is provided to government customers with limited rights as described in DAR 7-104.9A.

HISTORY

The MPX-32 Release 1.0 Reference Manual, Publication Order Number 323-001013-000, was printed June, 1979.

Publication Order Number 323-001013-100 (Revision 1, Release 1.3) was printed February, 1980.

Publication Order Number 323-001013-200 (Revision 2, Release 1.4) was printed July, 1980.

Publication Order Number 323-003663-000 (Revision 3, Release 1.5B) was printed September, 1982. The updated manual contains the following pages:

Title page
Copyright page
iii/iv through xiii/xiv
1-1 and 1-2
2-1 through 2-17/2-18
3-1 through 3-5/3-6
4-1 through 4-5/4-6
5-1 through 5-4
6-1/6-2
7-1 through 7-42
8-1 through 8-20
9-1 through 9-12
10-1 through 10-9/10-10
A-1 through A-7/A-8
B-1 through B-21/B-22
C-1 through C-39/C-40
D-1 and D-2
E-1 and E-2
F-1 through F-3/F-4
G-1 through G-3/G-4
GL-1 through GL-11/GL-12
IN-1 through IN-17/IN-18

CONTENTS

6. RECOVERING THE SYSTEM

7. SYSTEM GENERATION (SYSGEN)

8. THE MPX-32 SYSTEM DEBUGGER (H.DBUG)

ILLUSTRATIONS

TABLES

Documentation Conventions

Notation conventions used in command syntax and message examples throughout this manual are described below.

lowercase letters

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

```
!ACTIVATE taskname
```

means replace taskname with the name of a task, e.g.,

```
!ACTIVATE DOCCONV
```

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

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

means a break occurred on the specified task.

UPPERCASE LETTERS

In command syntax, uppercase letters specify a keyword must be entered as shown for input, and will be printed as shown in output. For example,

```
SAVE filename
```

means enter SAVE followed by a filename, e.g.,

```
SAVE DOCCONV
```

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

```
taskname,taskno ABORTED

*YOUR TASK IS IN HOLD. ENTER CONTINUE TO RESUME IT
```

Braces { }

Elements placed one under the other inside braces specify a required choice. You must enter one of the arguments from the specified group. For example,

```
{counter  }
{startbyte}
```

means enter the value for either counter or startbyte.

Brackets [ ]

An element inside brackets is optional. For example,

[CURR]

means the term CURR is optional.

Items placed one under the other within brackets specify you may optionally enter one of the group of options or none at all. For example,

[base name]
[progname]

means enter the base name or the program name or neither.

Items in brackets within encompassing brackets specify one item is required only when the other item is used. For example,

TRACE [lower address [upper address]]

means both the lower address and the upper address are optional, and the lower address may be used alone. However, if the upper address is used, the lower address must also be used.

Commas between multiple brackets within an encompassing set of brackets are semi-optional; that is, they are not required unless subsequent elements are selected. For example,

M.DFCB fcb,lfc [,[a],[b],[c],[d],[e]]

could be coded as

M.DFCB FCB12,IN

or

M.DFCB FCB12,IN,,ERRAD

or

M.DFCB FCB13,OUT,,ERAD,,PCK

Horizontal Ellipsis ...

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

name ,...,name

means you may enter one or more name values separated by commas.

Vertical Ellipsis :

The vertical ellipsis specifies commands, parameters, or instructions have been omitted. For example,

```
COLLECT 1
  .
  .
  .
LIST
```

means one or more commands have been omitted between the COLLECT and LIST commands.

Numbers and Special Characters

In a syntax statement, any number, symbol, or special character must be entered as shown. For example,

(value)

means enter the proper value enclosed in parentheses; e.g., (234).

Underscore

In syntax statements, underscoring specifies the letters, numbers or characters that may be typed by the user as an abbreviation. For example,

ACTIVATE taskname

means spell out the command verb ACTIVATE or abbreviate it to ACTI.

RESET

means type either RESET or RST.

In examples, all terminal input is underscored; terminal output is not. For example,

TSM > EDIT

means TSM was written to the terminal; EDIT is typed by the user.

Subscript Delta ▲

A subscript delta specifies a required space. For example,

EDT>STO▲TSSPGM

means a space is required between O and T.

1. BUILDING AND MAINTAINING THE MPX-32 SYSTEM - AN OVERVIEW

MPX-32 makes use of regular system utilities such as the File Manager and Text Editor plus a fully operational, interactive system to provide familiar mechanisms for building and maintaining resident operating systems. With MPX-32, the user can configure a resident system by running the System Generator utility, SYSGEN, just as he would run any other utility, interactively or in batch. SYSTEMS also supplies a System Debugger that can be used to debug a resident operating system or resident user-developed interrupt and device handlers.

This volume provides primary documentation on cold start/warm start installation from the SDT package (System Distribution Tape and Save Tape), SYSGEN, online and CPU front panel restart, the System Debugger, the System Patch facility, and Memory-Only MPX-32.

The MPX-32 operating system supports floppy disc usage. All references to the System Distribution Tape (SDT) apply whether the distribution medium is magnetic tape or floppy disc. SDT will also be used to refer to both SDT and the Save Tape.

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

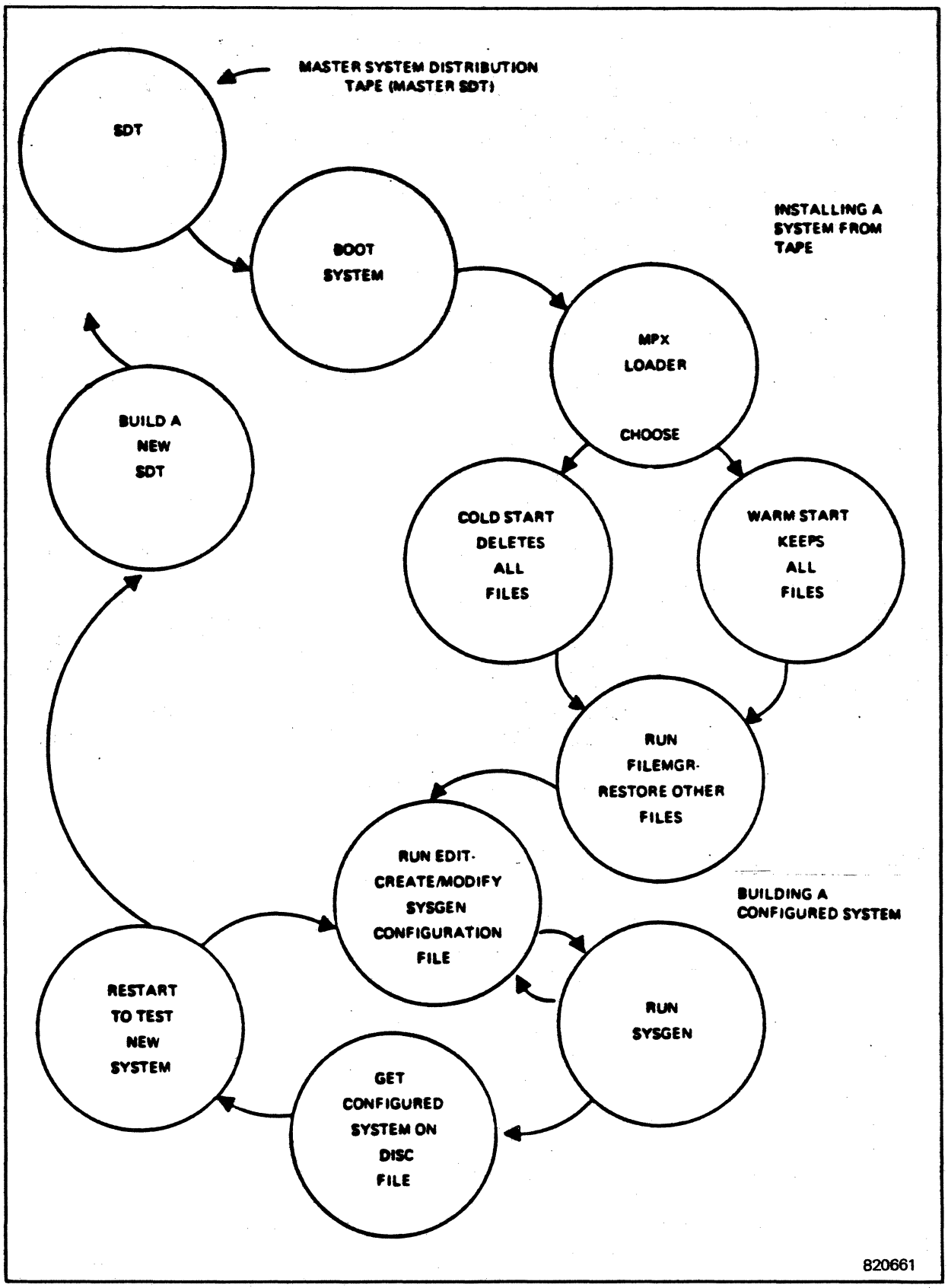

Figure 1-1
MPX-32
Installation/Configuration Overview

2. INSTALLING A STARTER SYSTEM

SYSTEMS supplies a starter system as the first file on the master System Distribution Tape (SDT). This section describes the minimum hardware configuration supported by the starter system, the format of the master SDT, and steps through the process of booting the starter system.

2.1 Hardware/Firmware Requirements

The following hardware is anticipated by the starter system for the installation process on a SYSTEMS 32/7x computer:

		Software	Hardware	
Memory-Only	Disc-Based		Channel	Subaddress
64KW Memory	128KW Memory			
Magnetic Tape (Class E or F)	Magnetic Tape (Class E or F)	1000	10	00
TLC Console	TLC Console	7E00	78	01
RTOM Interval Timer	RTOM Interval Timer	--	79	04
	Disc Drive (Class E or F)	0800	08	00
	Line Printer	7A00	78	02
	Card Reader	7800	78	00
RTOM Real Time Clock		--	79	06

The following hardware is anticipated by the starter system for the installation process on a CONCEPT/32 computer:

		Software	Hardware	
Memory-Only	Disc-Based		Channel	Subaddress
64KW Memory	128KW Memory			
XIO Magnetic Tape (Class F) -or-	XIO Magnetic Tape (Class F) -or-	1000	10	00
IOP Floppy Disc (2nd Floppy Disc)	IOP Floppy Disc (2nd Floppy Disc)	7EE0 7EE1	7E 7E	E0 E1
IOP Console	IOP Console	7EFC	7E	FC
	IOP Line Printer (2nd Line Printer)	7EE8 7EE9	7E 7E	E8 E9
	Disc Drive (Class F)	0800	08	00
IOP Interval Timer		--	7F	04
IOP Real Time Clock		--	7F	06

Table 2-1

Minimum Hardware Configuration

Only the OPCOM console is configured in the starter system. Other terminals should be included in the user configurations of the system created via SYSGEN. When terminals have been configured, they can be initialized (by a system module called J.TINIT) with parameters as follows until the user creates a terminal initialization file named LOGONFLE:

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

The following discs supported by MPX-32 can be used when booting from the master SDT:

Disc	Disc Code
4MB Fixed Head Disc - Class E Device	FE004
10MB Cartridge Module Disc - Class E Device	CE010
40MB Moving Head Disc - Class E Device	ME040
80MB Moving Head Disc - Class E Device	ME080
300MB Moving Head Disc - Class E Device	ME300
5MB Fixed Head Disc - Class E Device	FE005
1.2MB Floppy Disc - Class F Device	FL001
40MB Moving Head Disc - Class F Device	MH040
80MB Moving Head Disc - Class F Device	MH080
300MB Moving Head Disc - Class F Device	MH300
5MB Fixed Head Disc - Class F Device	FH005
32MB Cartridge Module Disc - Class F Device	CD032

2.1.1 Firmware Revisions

Model	Description	Wirelist/Board	Firmware
32/7x SERIES			
2000-4	CPU A	160-103177-001P	
2000-5	CPU B	160-103178-001M	
2000-6	CPU C System Control	160-103179-001V	531-322600-013
2142-1	Panel Interface	411-103022-008B	531-322451-003
2117	RTOM (Wire)	411-103079-002A	
2345	RTOM (Copper)	160-103109-001H	
2382	SCM MBC (Copper) High Speed Tape	160-103265-001F	531-322490-001
8050	Processor RPU	161-103149-001F	531-322606-001
8055	II DI Disc Processor	161-103282-001C	531-322621-004
9004	TLC (Wire)	411-103038-008B	531-322460-003
9005	TLC (Copper) Moving Head Disc Controller	160-103175-001G	531-322601-003
9010	(Wire) Magnetic Tape Controller	411-103023-007A	531-322464-003
9012	(Wire) Magnetic Tape Controller	411-103039-001A	531-322462-006
9013	(Copper) Magnetic Tape Processor	160-103122-001C	531-322462-006
9020	(Copper) Disc Processor	160-103210-001B	531-322607-005
9024-1	DI (Copper)	161-103282-001C	531-322621-004
9103	16MB GPMC (Wire)	411-103033-006B	531-322610-002
9104	GPMC (Wire) GPMC Terminator	411-103033-006B	531-322480-010
9105	(Wire)	411-103078-003A	
9110	ALIM (Wire)	411-103113-003B	531-322486-005
9122	ADS (Wire)	411-103025-007A	531-322465-001A
9131	HSD II (Copper)	160-103364-001B	531-322577-008

Note on Magnetic Tape: Earlier releases of the MTC will operate properly with earlier releases of the 70 SERIES CPU. However, Revision -5 and up of the MTC firmware require Revision -8 or greater CPU firmware levels.

Table 2-2

CPU Hardware/Firmware Revisions

Model	Description	Wirelist/Board	Firmware
32/27 SERIES			
	Single Slot CPU	160-103419-001F	531-322655-007
3021	Integrated Memory Module	160-103425-002B	531-322650-001
8000	Input/Output Processor (IOP)	160-103424-001D	531-322680-004
8000-1	IOP DI	160-103431-001E	
8030	Line Printer/ Floppy	160-103448-001B	531-322687-003
8050	High Speed Tape Processor RPU	161-103149-001F	531-322606-001
8050-1	High Speed Tape Processor DI	160-103369-001E	
8055	Disc Processor II DI	161-103282-001C	531-322621-004
8055-1	Disc Processor II RPU	161-103351-001C	531-322626-002
8055-2	Disc Processor II PROM	161-103365-001	531-322627-001
8191	Disc Processor 8DR (Copper)	160-103291-001A	
8510	8-Line Async (Copper)	160-103445-001B	531-322685-002
8580	RS232 Dest. (Copper)	160-103434-001	
9020	(Copper)	160-103210-001B	531-322607-005
9131	Disc Processor HSD II (Copper)	160-103364-001B	531-322577-008
32/87 SERIES			
Data specifics are not known at this time.			

Table 2-2

CPU Hardware/Firmware Revisions
(continued)

2.1.2 Other Requirements

In addition, the MPX-32 starter system requires:

- o upgraded firmware for IOM tape drive controllers/formatters
- o a copper RTOM with the interval timer set for interrupt at 38.4 microseconds; the interval timer must be jumpered at hardware device address 7904, interrupt level 7F.
- o copper TLC

2.2 The Master System Distribution Tape (SDT)

Figure 2-1 describes the layout of the master SDT. The master SDT structure is identical to a user SDT, except that in order to support varying discs for the initial MPX-32 resident image, the master SDT contains a system dispatcher at the beginning that routes processing to one of several minimum starter systems on the master tape.

After the system dispatcher, an SDT contains the resident operating system and the MPX-32 initialization loader BOOTxx followed by key load modules for MPX-32 system tasks. These modules are required to support a fully operational system which is then used to restore utility processors such as the Cataloger, libraries, and other files provided by SYSTEMS on the Save Tape which is part of the SDT package.

Table 2-3 lists the software included on the master SDT.

Note: Due to the number of starter systems available, one tape is used as an SDT. A second tape, a Save Tape, contains the full set of processors and other files that make up the non-resident portion of MPX-32.

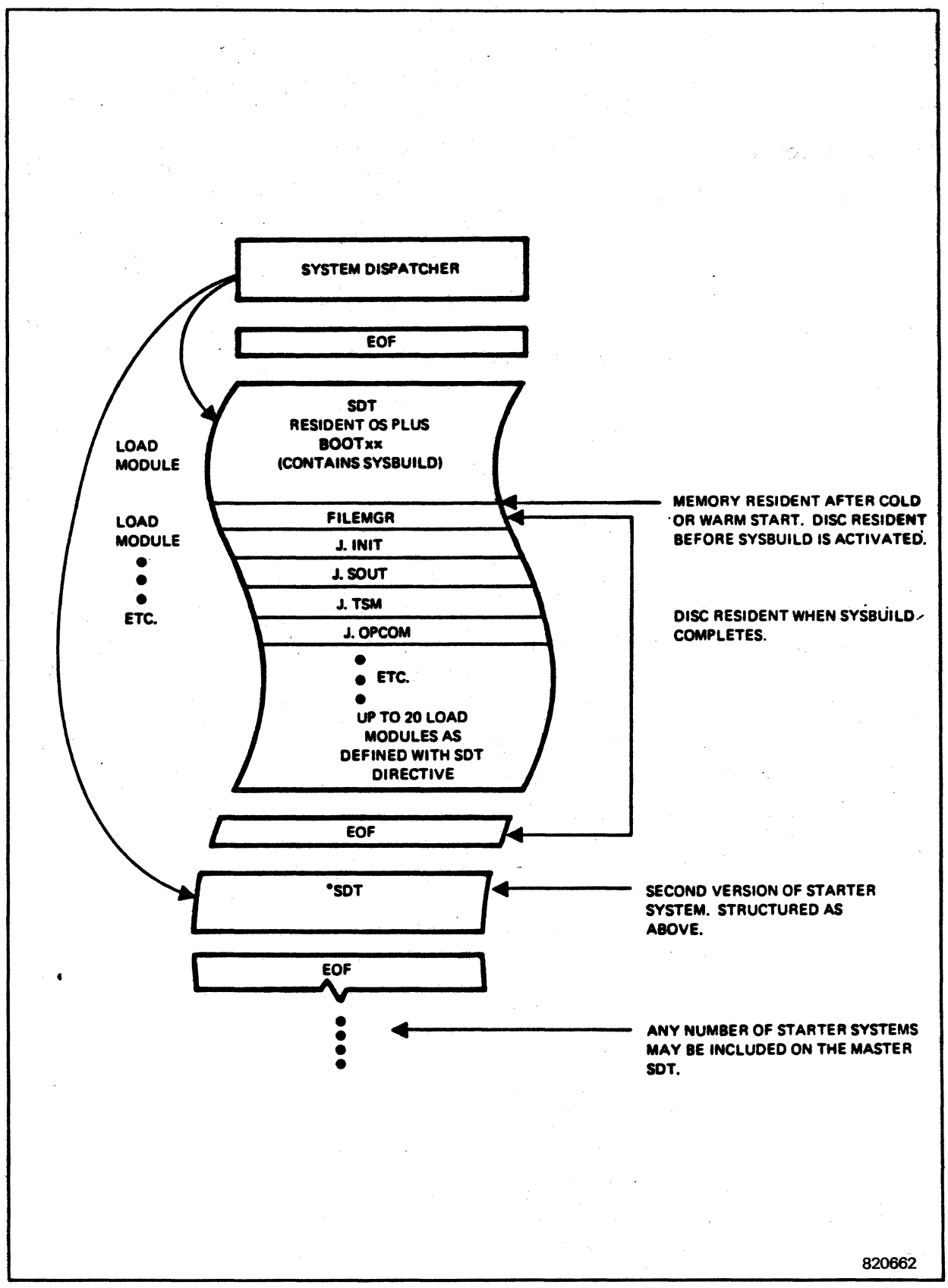

Figure 2-1. Master System Distribution Tape Format

Table 2-3

Deliverable Software for MPX-32

Module Name	Description	RES OS	Object File	Load Module	Using Library**
H.EXEC	CPU Scheduler	x	OH.EXEC		
H.MONS	Services	x	OH.MONS		
H.IOCS	I/O Control System	x	OH.IOCS		
H.FISE	File System Exec.	x	OH.FISE		
H.ALOC	Allocator Allocator (CONCEPT/32)	x	OH.ALOC* OH.ALOC2*		
H.LODR	Loader	x	OH.LODR		
H.SOUT	System Output	x	OH.SOUT		
H.TSM	Terminal Manager-Task Activation	x	OH.TSM		
H.DBUG	System Debugger for Resident OS	x	OH.DBUG		
H.IPnn	Interrupt/Trap Handlers	x	OH.IPnn		
H.ICP	Indirectly Connected Timers	x	OH.ICP		
H.ADIO	Analog Digital Handler	x	OH.ADIO		
H.CD00	Cartridge Disc Handler	x	OH.CD00		
H.SLMP	SLIM Handler	x	OH.SLMP		
H.FLIOP	IOP Floppy Disc Handler	x	OH.FLIOP		
H.A8IOP	IOP 8-Line Async Handler	x	OH.A8IOP		
H.F8IOP	Full Duplex 8-Line Async	x	OH.F8IOP		

*Memory-only systems require MEMONLY version
**M.MPXMAC is the macro library used for any Assembler routine unless otherwise specified

Module Name	Description	RES OS	Object File	Load Module	Using Library**
H.CTIOP	Console Handler (IOP)	x	OH.CTIOP		
H.DF01	Fixed Head Disc	x	OH.DF01		
H.DM01	Moving Head Disc	x	OH.DM01		
H.DP01	Extended I/O Moving Head Disc - Phase I	x	OH.DP01		
H.DP02	Extended I/O Moving Head Disc - Phase II	x	OH.DP02		
H.ASMP	GPMC ASYNC Comm. for Terminals	x	OH.ASMP OH.QASMP***		
H.CR00	Card Reader	x	OH.CR00		
H.CT00	TLC OPCOM Console	x	OH.CT00		
H.LP00	Line Printer	x	OH.LP00		
H.CPMP	GPMC Card Reader-Punch	x	OH.CPMP OH.QCPMP***		
H.MT00	Mag Tape	x	OH.MT00		
H.TY10	ADS Terminal Interface	x	OH.TY10		
H.EXIO	Extended I/O Multiplexer	x	OH.EXIO		
H.MUX0	GPMC Multiplexer	x	OH.MUX0 OH.QMUX0***		
H.PTMP	GPMC Paper Tape Reader - Punch	x	OH.PTMP OH.QPTMP***		
H.BSMP	Bisynchronous Line Interface Module	x	OH.BSMP OH.QBSMP***		
H.XMT	Extended I/O Mag Tape	x	OH.XMT		
H.IPU	CPU to IPU Trap Processor	x	OH.IPU		
H.CPU	IPU to CPU Trap Processor	x	OH.CPU		

*Memory-only systems require MEMONLY version
**M.MPXMAC is the macro library used for any Assembler routine unless otherwise specified
***For Pre 1.4 GPMC structure

Module Name	Description	RES OS	Object File	Load Module	Using Library***
H.IOPX	IOP Channel Executive	x	OH.IOPX		
H.IPUIT	IPU Accounting	x	OH.IPUIT		
H.LPIOP	IOP Line Printer	x	OH.LPIOP		
H.GPMCS	GPMC Subroutines (1.4)	x	OH.GPMCS		
H.IOC11	GPMC Status Routine (Pre 1.4)	x	OH.IOC11		
J.SWAPR	Swapper Swapper (CONCEPT/32)	x	OH.SWAPR* OH.SWAP2*		
J.INIT	Disc Initialization		OJ.INIT	J.INIT	
J.TINIT	Terminal Initialization		OJ.TINIT	J.TINIT	
J.JOBC	Job Control		OJ.JOBC	J.JOBC	
J.SOUT	Output Spooling		OJ.SOUT	J.SOUT	
J.SSIN	Input Spooling-Devices		OJ.SSIN	J.SSIN	
J.SSIN2	Input Spooling-Files			J.SSIN2	
J.TSM	Terminal Control		OJ.TSM	J.TSM	
J.RTMCT	RTM Cataloger for MPX		OJ.RTMCT	RTMCATL	
J.PRJCT	Project Accounting		OJ.PRJCT	J.PRJCT	N/A
J.ACCNT	Accounting Utility		OJ.ACCNT	J.ACCNT	N/A
OPCOM	Operator Communications		OJ.OPCOM	OPCOM	
OPCMM	Operator Communications on Memory-only System		OJ.OPCOM*	OPCMM	
FILEMGR	File Manager		OJ.FMGR	FILEMGR	
ASSEMBLE	Assembler		OJ.ASSM	ASSEMBLE	
CATALOG	Cataloger		OJ.CATL	CATALOG	

*Memory-only systems require MEMONLY version
**M.MPXMAC is the macro library used for any Assembler routine unless otherwise specified

Module Name	Description	RES OS	Object File	Load Module	Using Library**
COMPRESS	Object Code Concatenator		OJ.COMP	COMPRESS	
DEBUG	Task Debugger		OJ.DEBUG	MPXDB	
DPEDIT	DATAPOOL Editor		OJ.DPED	DPEDIT	M.MACLIB
EDITOR	Text Editor		OJ.EDIT	EDIT	
ERR?	Error Message Expansion		OJ.ERR?	ERR?	
KEY	M.KEY File Editor		OJ.KEY	KEY	
LIBED	Subroutine Library Editor		OJ.LIBED	LIBED	M.MACLIB
MPXLIB	MPX System Subroutine Library		MPXLIB (null file)		
MPXDIR	Associated Directory		MPXDIR (null file)		
MACLIBR	Macro Library Editor		OJ.MACLE	MACLIBR	N/A
M.MPXMAC	MPX Macro Library		SJ.MPXMC (source)		
M.MACLIB	MPX/RTM Macro Library (Equates for RTM/MPX)		SJ.RTMMC (source)		
M.RTMMAC	RTM Macro Library (RTM use only)		SJ.MCRTM (source)		
MEDIA	Media Conversion		OJ.MEDIA	MEDIA	M.MACLIB
RESTART	On-Line Restart		OJ.REST	RESTART	
UPDATE	Source Update		OJ.UPDAT	UPDATE	M.MACLIB
SYSGEN	System Generator		OJ.SEXEC	SYSGEN	SG.LIB
SYSGEN	Initialization Overlay		OJ.SINIT	S.INIT	SG.LIB
SYSGEN	Overlay 1		OJ.SPH01	S.PH01	SG.LIB
SYSGEN	Overlay 2		OJ.SPH02	S.PH02	SG.LIB
SYSGEN	Overlay 3		OJ.SPH03	S.PH03	SG.LIB

*Memory-only systems require MEMONLY version
**M.MPXMAC is the macro library used for any Assembler routine unless otherwise specified

Name	Description	RES OS	Object File	Load Module	Using Library*
SYSGEN	Overlay 4		OJ.SPH04	S.PH04	SG.LIB
BOOT7X	MPX Initializer for 32/7x		OJ.BOOT7	BOOT7X	
BOOT27	MPX Initializer for CONCEPT/32		OJ.BOOT2	BOOT27	
BOOTMEMO	MPX Initializer for Memory-only MPX-32			BOOTMEMO	

Module Name	Description	Source File
N/A	Sample SYSGEN for Small Systems	SG.S
N/A	Sample SYSGEN for Medium Systems	SG.M
N/A	Sample SYSGEN for Large Systems	SG.L

Job Stream	Description
JJ.SYSGN	Job control to Assemble, Libed, and Catalog SYSGEN
SJ.STBLS	Allows addition of new devices to SYSGEN tables

COMPRESS File Assignments	Description
JH.7X	IN assignment for 32/7X disc-based system
OH.7X	OT assignment for 32/7X disc-based system
JH.7XM	IN assignment for 32/7X memory-only system
OH.7XM	OT assignment for 32/7X memory-only system
JH.27	IN assignment for 32/27 or 32/87 disc-based system
OH.27	OT assignment for 32/27 or 32/87 disc-based system

*M.MPXMAC is the macro library used for an Assembler routine unless otherwise specified

COMPRESS File Assignments	Description
JH.27M	IN assignment for 32/27 memory-only system
OH.27M	OT assignment for 32/27 memory-only system

2.3 Booting the Resident System and System Builder

Mount the master System Distribution Tape on the drive at channel 10, subaddress 00. Turn on the drive.

Mount an initialized disc pack on the drive at channel 08, subaddress 00. Turn on the drive.

Note: If the disc pack is not initialized, run appropriate disc diagnostics to initialize it before continuing. This ensures the sector addresses are verified, surfaces are in good condition, etc.

At the CPU front panel:

Depress: SYSTEM RESET
KEYBOARD tapeaddress (Enter 1000)
IPL

The system dispatcher from the master SDT prompts you to enter the name of the desired system image. With the master SDT, the starter system image is described by the type of disc you are using to store the resident operating system. This is equivalent to defining a disc later as the System Master Directory (SMD) disc and is used by the dispatcher to route processing to a version of the starter system on the SDT that accommodates that particular disc in the minimum configuration:

DESIRED SYSTEM IMAGE: devcode

The desired image is based on the device codes (CE010, MH300, etc.) shown in Section 2.1. A similar set of codes is used to describe discs with the SYSGEN DEVICE directive.

The appropriate starter system is loaded from the master SDT and a memory-resident system is built by the BOOT7X or BOOT27 program, which is included in the starter system. If you are not booting from a master SDT, these messages do not appear.

Note: If NOTHING happens, check the hardware/firmware revisions of your system (see Table 2-2) and contact your SYSTEMS Field Representative.

2.3.1 Control Switches

After the IPL button has been pushed, various initialization processes can be inhibited or enabled by setting the appropriate Control Switches. The assignment of the 13 switches is as follows:

Switch	Function (if switch is set)
0	Enables checksumming of the disc allocation map(s) and controls whether FS04, FS05, and FS06 errors are reported. If the switch is not set, error conditions are ignored.
1	J.INIT starts System Debugger before processing patches
2	Inhibits patch processing (see Chapter 9, Section 9.3)
3	Inhibits terminal initialization
4	Inhibits accounting functions
5-12	Reserved

The Control Switches can be accessed via the CPU front panel or from the IOP console. To set, for example, switch 0 from the front panel, the following buttons are pushed:

7x Control Panel	IOP Console
KEYBOARD	CS=80000000
80000000	
WRITE	
CSWS	

2.4 Resident System Cold Start/Warm Start Processes

A cold start from the SDT initializes the System Master Directory (SMD). This must be done the first time you boot the starter system. A cold start is also used to change either the size or the location of the SMD via the SYSGEN SMD directive. Cold start can also be used if you want to clear any traces of a previous system when installing an SDT. Cold start deletes and recreates the SMD. All files which previously existed are thus deleted.

If a moving head disc is added to the system with a device address lower than any of those present, the internal device tables of MPX-32 will be modified, thereby invalidating the file pointers in the SMD. As an example, device DM0802 can be warm started into a system where DM0800 is previously defined. However, DM0400 cannot.

If a fixed head disc is added during a warm start, it will also invalidate the SMD file pointers.

A warm start from the SDT reloads the resident operating system from tape, but does not alter the SMD. All files which existed previously remain intact.

Whether cold or warm start is used, loading from the SDT clears any M.SID and M.SOD directories if they exist, and any spooled batch files are lost.

```
COLD OR WARM START (C/W)?
```

Enter C for cold start, W for warm.

2.5 The System Builder

After BOOTxx finishes the cold or warm start, it activates a task called SYSBUILD, which is part of BOOTxx. At this point, you can either get other central load modules such as FILEMGR, J.INIT, etc, from the SDT or leave that part of the system as it was before you started installing the SDT. (This is useful if you want to only patch the OS.)

The first time you install MPX-32, you will want to resume reading modules from the SDT:

```
TASK SYSBUILD, 02000001 MOUNT SCRA VOL ON M91000 DEV,R,A,H?
```

Enter R (Ready) to continue installing modules from the SDT.

Enter A (Abort) to avoid installing subsequent SDT files.

Do not enter H (Hold) or supply a device mnemonic in response to the MOUNT message when installing a system (cold start or warm start).

If you resume the tape, modules are restored from the SDT up to an EOF written by FILEMGR. See Section 4.3. If you abort the tape mount request, only the resident image is written to disc.

After modules are loaded, a DISMOUNT message is displayed and J.INIT is activated as a task. J.INIT displays SYSTEMS MPX-32 to indicate that key modules are in operation. It proceeds to reinitialize disc allocation maps and deallocates temporary file space as needed. There is a delay during this stage of the installation process.

```
DISMOUNT SCRA VOL ON M91000
SYSTEMS MPX-32 *systemname*
```

After disc map reinitialization is complete, J.INIT displays a verification message and activates J.TSM.

```
INITIALIZATION COMPLETE
```

J.TSM then activates OPCOM, which prompts you to enter the current date and time.

```
ENTER DATE AND TIME
??
```

2.5.1 Memory-Only System Initialization

System initialization of a memory-only MPX-32 (MPX-32/M) system commences when control is transferred to the BOOTMEMO module after IPL from a SDT. At this point, the system image specified on the IPL medium (magnetic tape or floppy disc) has been loaded into memory followed by the BOOTMEMO module. If the System Debugger has been included as part of the system image, a prompt will appear at the operator's console. In this way, system modifications can be made prior to system initialization and task loading. Action is then taken by BOOTMEMO to initialize the interrupt/trap mechanism and create SYSBUILD as a ready-to-run task. The system commences execution within the context of the task SYSBUILD which loads tasks into memory sequentially from the IPL medium until an end-of-file (previously written by the File Manager) is encountered.

SYSBUILD reads the load module preamble and calls the pre-activation services of the system allocator on behalf of the task. This results in the construction of a pseudo-TSA and DQE entry for the task and places it on the ready-to-run queue. SYSBUILD then suspends itself allowing the system scheduler to select the task for further activation by the allocator. At this time, the remainder of the task is loaded sequentially from the IPL medium, and a fully independent task is established. The system allocator then forces each task to resume SYSBUILD and suspend itself as the last step in the activation process. This suspend/resume cycle continues between SYSBUILD and each task to be loaded from the SDT until an end-of-file mark is encountered when attempting to read the next preamble.

After all tasks have been loaded from the SDT, SYSBUILD issues a completion message and prompts the user to enter the system date and time (provided an operator's console has been configured). SYSBUILD then directs a resume request to all user tasks in the order they were loaded from the SDT. These tasks have completed the activation sequence and will execute in a manner dictated by the system scheduler, when they are resumed.

The final action of SYSBUILD is to deallocate the IPL unit and perform a normal system exit. The system is now functioning within a memory-only environment comprised exclusively of resident tasks and an optional operator communications facility.

If any errors occur during the task loading and activation sequence, SYSBUILD will abort with a message, halting the installation process. The task that caused the error will have to be corrected and the installation procedure repeated.

2.6 Operating Under the Starter System

You now have an MPX-32 system that is both disc and memory resident.

OPCOM is active on the OPCOM console and can be used as documented in Chapter 4, Volume 1. The File Manager is also included as a central system load module. It is used to move other processors and system files from the Save Tape, that is part of the SDT package, or another tape to disc. The File Manager is documented in Volume 2, Chapter 6.

For OPCOM use on a memory-only system, see Volume 3, Chapter 10.

2.7 Restoring Utility Processors, Libraries, and Other Files from the Master Save Tape

Load modules for system utilities such as the Assembler, the Editor, SYSGEN, etc., are saved to a Save Tape as part of the SDT package using regular FILEMGR SAVE commands and are restored by using FILEMGR RESTORE commands. Each SAVE command allows 20 sets of files maximum and an end of file (EOF) is written after each group specified with SAVE. In restoring processors and other key system files, a simple RESTORE command is used for each group. Initially you will want to restore all groups of files from the Save Tape. In subsequent interactions, all selective restore capabilities of the FILEMGR are available for system installation and maintenance. The process outlined below is used when installing a starter system.

OPCOM is up on the OPCOM console. No other terminals are yet available. At the OPCOM console access TSM. Activate the FILEMGR to restore the first group of load modules. Before using the FILEMGR, assign the Save Tape for input.

```
??EXIT
TSM > ASSIGN3  IN=M91000,SDT
TSM > FILEMGR
FIL > RESTORE
FIL > RESTORE
        .
        .
        .
```

The number of groups of files saved can vary, and an equivalent number of RESTORE commands are required to install them. A SAVELOG can be performed on the Save Tape to list the files and groups of files that are contained on the tape. Restore all groups of utilities, libraries, and system files (one RESTORE command per group).

2.8 Example

First use of master SDT on user system. An initialized disc is located on DM0800; the master SDT is mounted on tape drive M91000.

CPU Front Panel	SYSTEM RESET KEYBOARD 1000 IPL	IOP Console	RST IPL=1000

OPCOM console

```
DESIRED SYSTEM IMAGE: ME080ME7
COLD START OR WARM START (C/W): C
TASK SYSBUILD, 02000001 MOUNT SCRA VOL ON M91000
DEV,R,A,H? R
DISMOUNT SCRA VOL ON M91000
SYSTEMS MPX-32  *ME080ME7*
INITIALIZATION  COMPLETE
```

(*ME080ME7* is the file name used on the master SDT to identify a particular starter system image. If booting from a user SDT, the name of the user file produced at SYSGEN and specified as the sysfile with the SDT command is displayed.)

```
ENTER DATE AND TIME
??month/day/year,hour:minute:second
??EXIT
TSM > ASSIGN3 IN=M91000,SAVE
TSM > FILEMGR
FIL > RESTORE
FIL > RESTORE
      .
      .
      .
FIL > EXIT
```

3. BUILDING AND TESTING A USER-CONFIGURED SYSTEM

This section describes how to SYSGEN and test a resident system geared to a specific installation. System generation via the SYSGEN utility is fully described in Section 7. This section is designated only to establish SYSGEN in the cycle of building an MPX-32 system.

3.1 Building the SYSGEN Input Files

3.1.1 Building the Directive Input File

The directive input file for SYSGEN determines the configuration of a user system: what hardware to support, what interrupts to connect, what devices to use for spooled system I/O, etc. On the master SDT, SYSTEMS provides three files, any of which can be used either as a model or as a working base for preparing a SYSGEN directives file:

File	Description
SG.S	Sample directive input file for a Small System
SG.M	Sample directive input file for a Medium System
SG.L	Sample directive input file for a Large System

The Editor (Volume 2, Chapter 5) is used to build or modify the SYSGEN directives file. Assuming that a starter system and MPX-32 processors have just been installed as described in the previous section, select and edit one of the samples or build your own directive input file from scratch, remembering the resultant file must be stored not saved via the Editor. The samples provide a model for building a system in the sense that progressing from a small to large configuration may facilitate checkout and test of a final configuration.

The System Debugger (DBUG) can be included in the user's resident system via a SYSGEN PROGRAM directive and this is recommended, particularly in an initial system. The System Debugger adds approximately 4 KW to the size of the resident system.

3.1.2 Building the Object Input File

Using the directive input file, SYSGEN determines which system modules, user modules, interrupt handlers and trap handlers it will need to build the target system. It reads the object for these modules and handlers from the object input file. On the master SDT, SYSTEMS provides one object input file for each of the supported machine types:

File	Description
OH.7X	Object input file for a 327X system.
OH.7XM	Object input file for a 327X memory-only system.
OH.27	Object input file for a 3227 or 3287 system.
OH.27M	Object input file for a 3227 memory-only system.

If you wish to add your own object modules to one of the above object input files, SYSTEMS provides a task called COMPRESS described in the next section.

3.1.3 The COMPRESS Task

The COMPRESS task will build a file composed of any number of object files. Its input file (logical file code 'IN' whose default assignment is to the stored file JH.7X) must contain the ASCII names, one per record/line, of the object modules to copy to the output file. The output file (logical file code 'OT') is defaulted to the file OH.7X.

COMPRESS outputs a control stream informing the user which files have been copied and how many records they contained. It also reports any allocation or read errors. The default listed output file ('LO' assignment) is to UT.

Syntax:

COMPRESS

Each of the provided object input files used by the SYSGEN task were created by using the COMPRESS task. The COMPRESS input file for each SYSGEN object input file is included on the master SDT and can be used as a model or a working base for modifying the SYSGEN object input files. The Swapper (OH.SWAPR) must remain as the last ASCII name in the COMPRESS input file.

COMPRESS 'IN' Assignment	COMPRESS 'OT' Assignment/ SYSGEN 'OBJ' Assignment
JH.7X	OH.7X
JH.7XM	OH.7XM
JH.27 (for CONCEPT/32)	OH.27 (for CONCEPT/32)
JH.27M	OH.27M

Example:

```
TSM>ASSIGN1  IN=JH.7X          (default assignment)
TSM>ASSIGN1  OT=OH.7X          (default assignment)
TSM>ASSIGN4  LO=UT             (default assignment)
TSM>COMPRESS
FILE OH.IP00 COPIED 2 RECORDS
FILE OH.IP01 COPIED 2 RECORDS
        .
        .
        .
TSM>
```

3.2 Running SYSGEN

Interactive and batch access, required and default assignments, and other aspects of running SYSGEN are covered in Chapter 7. One simple path for configuring a system is described in this section.

The logical file code (lfc) for a SYSGEN directives file is DIR. A modified version of SG.S, SG.M, or SG.L can be assigned to DIR. One of the object input files listed in Section 3.1.2 can be assigned to OBJ, which is the logical file code for the SYSGEN object input file. A TSM ASSIGN command can be used to assign the input files as shown below.

```
TSM > ASSIGN1 OBJ=OH.7X    (default assignment)
TSM > ASSIGN1 DIR=SG.X
TSM > SYSGEN
```

Or a job file can be used to run SYSGEN in batch with the above file assignments.

The SYSGEN output file which contains the resident operating system is specified via the SYSTEM directive in the SYSGEN directives file and is created automatically by SYSGEN, i.e., the user need not create a file space for the output file before running SYSGEN. The file name used with the SYSTEM directive is also the name to use as the system load file when you are ready to build a SDT.

3.3 Testing a SYSGEN'd System

MPX-32 allows the user to perform a one-shot restart of a SYSGEN'd system online via the TSM RESTART command. Initially, any owner is permitted to use this command. The M.KEY file can be used to prohibit use of RESTART by ownername.

Once a system is up and running with terminals configured and various activities in process, system restart is not an innocuous operation. However, during an initial installation process the user probably does not need to be concerned with other aspects of system use (the OPCOM console is the only interactive device configured). The user can simply issue the RESTART command to TSM when he is ready to test a system.

```
TSM > RESTART sysfile
```

Supply the name of the file specified with the SYSGEN SYSTEM directive and the SYSGEN'd configuration of the resident operating system will be booted from the disc file you name. If there are any problems in the file you attempt to boot, RESTART automatically defaults back to the starter system that was booted from the master SDT. (For restart defaults at subsequent evolutions of the user-configured operating system, see Chapter 5.)

Restart from disc file is essentially the same as a warm start from a SDT. Note that for either warm start or restart, the size or location of the System Master Directory (SMD) cannot be changed. In the SYSGEN directives file, do not change the SMD directive to a different disc other than the disc supplied for the starter system until you are ready to SYSGEN a tested system and create a user SDT for cold start.

If a moving head disc is added to the system with a device address lower than any of those present, the internal device tables of MPX-32 will be modified, thereby invalidating the file pointers in the SMD. As an example, device DM0802 can be warm started into a system where DM0800 is previously defined. However, DM0400 cannot.

If a fixed head disc is added during a warm start, it will also invalidate the SMD file pointers.

The RESTART command with no system file specified or IPL from the CPU front panel (see Chapter 5) causes the default system (in this case, the starter system) to return as the working resident system. You can continue to refine and develop the system configuration you require via the Editor, SYSGEN, and one-shot RESTART as needed. Figure 3-1 provides an overview of the online RESTART and test process.

3.4 Terminal Initialization and System Protection

Volume 1 Chapter 5 describes how to initilize terminals on MPX-32 and Volume 2 Chapter 7 describes how to build an M.KEY file to provide authorized owners access to the system and implement privileges by owner.

Using either capability is optional, particularly at this stage of building a system.

The RESTART process initializes terminals and puts M.KEY privileges into effect if the user has developed terminal initialization and M.KEY files.

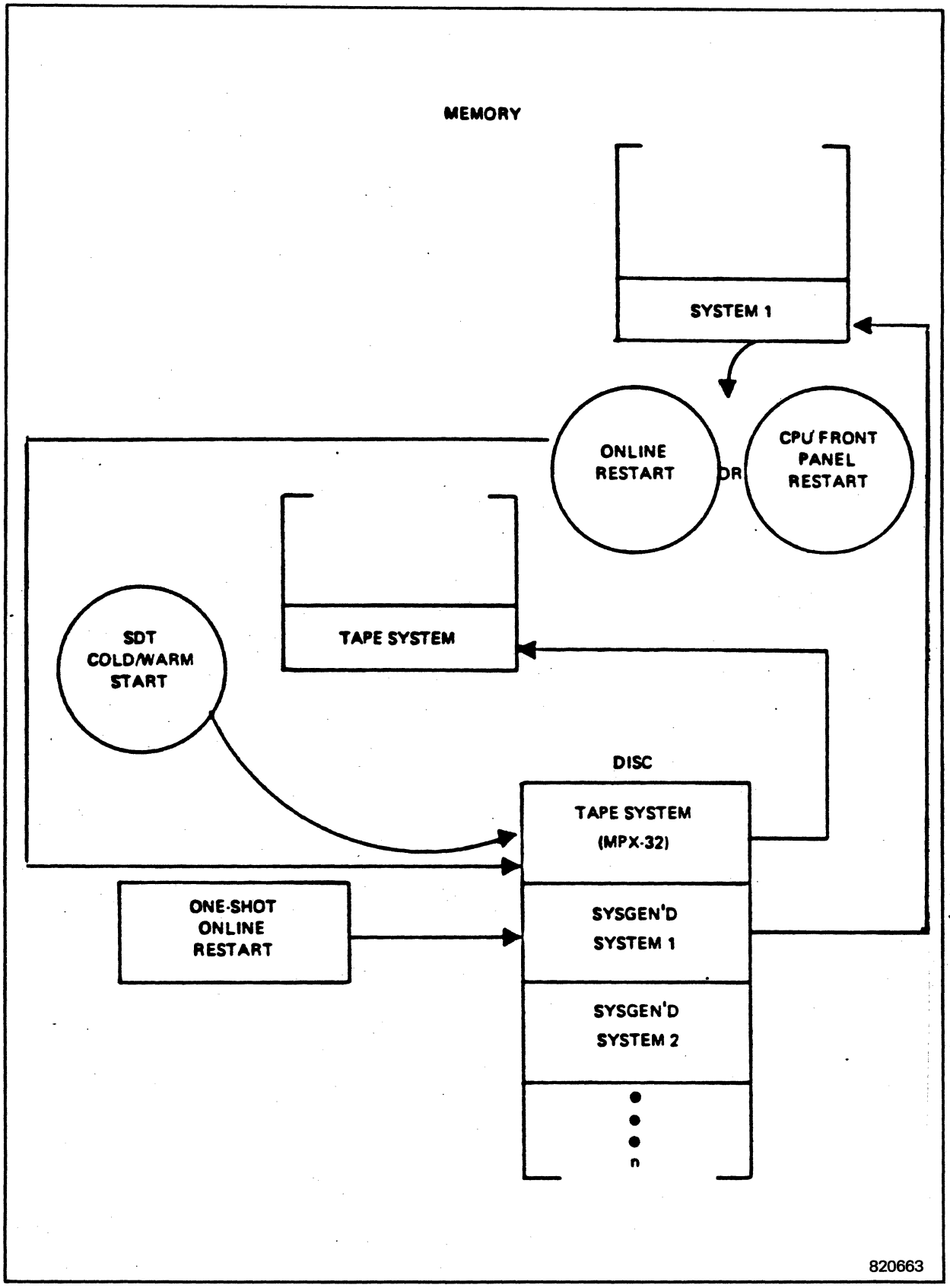

Figure 3-1. Testing a User-Configured System

4. INSTALLING A USER-CONFIGURED SYSTEM

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

4.1 System Master Directory (SMD) Changes

If you need to change the size of the SMD or its location, the changes must be done via a SDT cold start. Online RESTART will reject a change in the SMD location or size. The SYSGEN SMD directive is used to specify the disc on which the SMD is located.

If a moving head disc is added to the system with a device address lower than any of those present, the internal device tables of MPX-32 will be modified, thereby invalidating the file pointers in the SMD. As an example, device DM0802 can be warm started into a system where DM0800 is previously defined. However, DM0400 cannot.

If a fixed head disc is added during a warm start, it will also invalidate the SMD file pointers.

4.2 Creating a User System Distribution Tape (SDT)

The format of a user SDT is identical to the master SDT, with the exception that it will contain only one resident operating system (the one you have configured) and will have no system dispatcher. The key load modules restored with the resident system can also be increased to 20 maximum. Figure 4-1 describes the layout of load modules on a user SDT.

The minimum recommended set of key load modules for an SDT is:

- FILEMGR
- OPCOM
- J.INIT
- J.TSM
- J.SOUT

The FILEMGR LOG command can be used to obtain a list of all files that are on the system.

The FILEMGR SDT directive is used to specify key load modules; regular SAVE directives are used to build the rest of the files on a SDT. The SDT directive checks to ensure that each file specified is a valid load module.

Note: Because the Boot programs cannot process the multi-volume header, the SDT cannot be multi-volume.

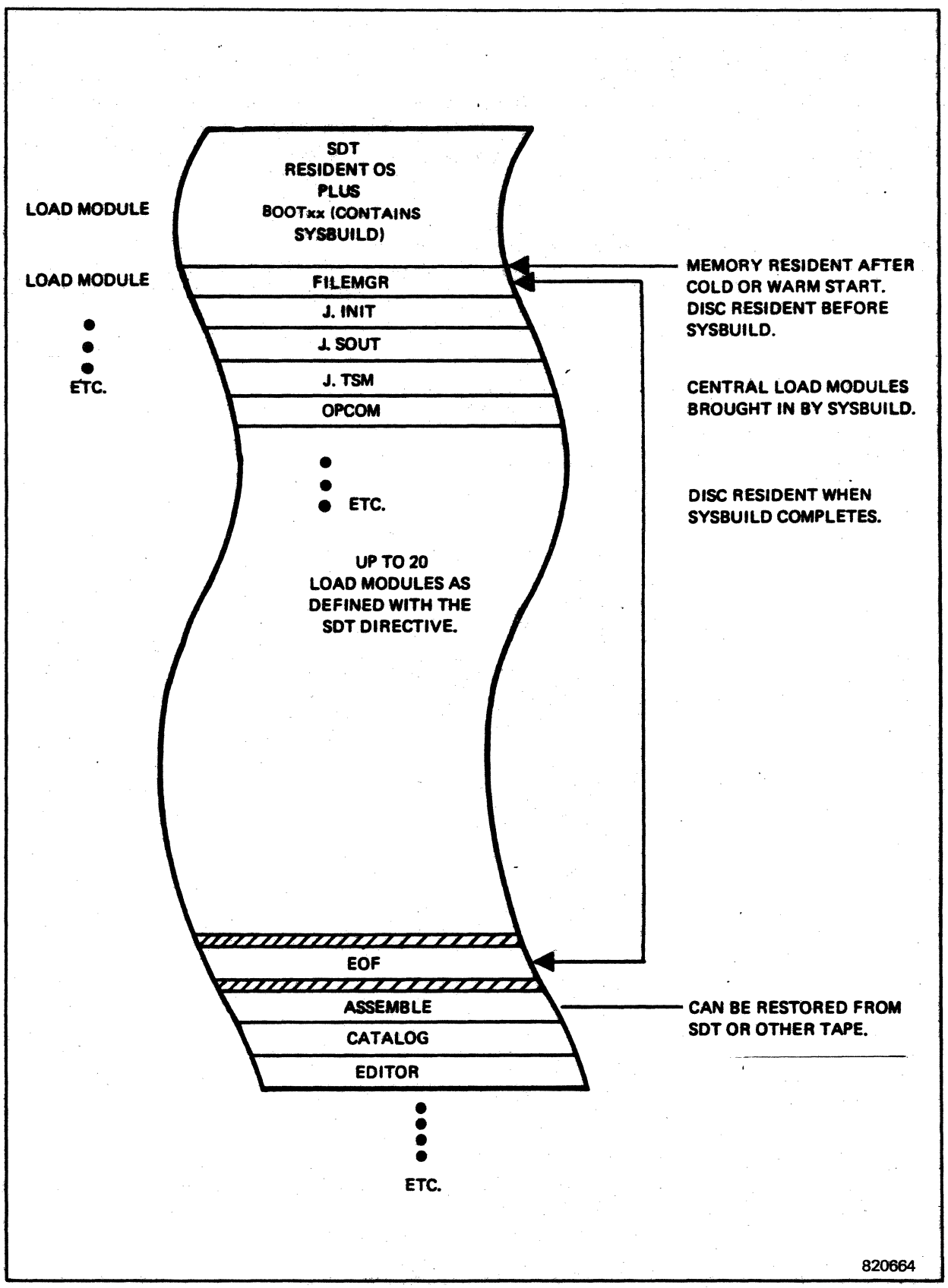

Figure 4-1

User System Distribution Tape Format

4.3 The SDT Directive

Key load modules are specified with the SDT directive.

Syntax:

SDT sysfile,BOOTxx [,loadmod] ,...

where:

sysfile — specifies the name of the load module that contains the resident operating system. You will have run SYSGEN to create a tailored version of the operating system and will specify the name of the output file generated with SYSGEN as the first load module. (Supply the file name used with the SYSGEN SYSTEM directive.)

BOOTxx — specifies the SYSTEMS-supplied load module, BOOT7X or BOOT27, which loads the resident system onto disc. BOOTxx also provides restart logic on disc for reloading from disc. It is required for all configurations of the SDT.

loadmod — specifies key load modules required to get a system that the user can communicate with.

For MPX-32, absolute minimum load modules are: FILEMGR, OPCOM, J.INIT, J.SOUT, and J.TSM. These are followed by an EOF written by the FILEMGR after the last load module specified with SDT.

For memory-only MPX-32 (MPX-32/M) systems, the SDT is built as follows:

SDT sysfile,BOOTMEMO [,OPCMM] [,loadmod] ,...

where:

sysfile — specifies the name of the load module that contains the resident operating system created by running SYSGEN. This is the file name used with the SYSGEN SYSTEM directive.

BOOTMEMO — specifies the SYSTEMS-supplied load module, BOOTMEMO, which performs the necessary system initialization and creation of the task SYSBUILD which in turn performs the loading and activation of user tasks supplied on the SDT.

OPCMM — specifies the memory-only operator communications task. This module is created by assembling the source module SJ.OPCOM into the object module OJ.OPCOM under the user name MEMONLY, with the memory-only conditional assemble flag (C.MEMO) set with the MPX-32 macro library. The resulting object module is then cataloged as the load module OPCMM so as not to erase the MPX-32 operator communications task (OPCOM). This module is optional, but if desired, it must be loaded from the SDT.

loadmod — specifies the load module names of user tasks to be loaded from the SDT and activated after system initialization. Up to 18 tasks (17 if OPCMM is specified) can be included in the SDT directive for immediate loading and activation. If OPCMM is specified, additional tapes or floppy discs with up to 20 tasks each can be generated for activation (see Section 10.5).

4.4 Saving System Processor and Utility Load Modules

Additional load modules, libraries, and files are saved and restored using regular FILEMGR directives.

Syntax:

SAVE FILE = filename [,filename] ,...

4.5 Installing a User SDT

All devices specified in the SYSGEN configuration of the user's system should be connected. All disc drives that are configured should contain initialized disc packs. The procedure described in Section 2.3 can be followed with these exceptions:

- The disc channel and subaddress for the disc that will contain the SMD does not have to be 0800. The disc used for the resident system booted from tape is the disc specified with the SYSGEN SMD directive. The channel on which it is configured is also described at SYSGEN.
- The system dispatcher is not on the user SDT and you are not prompted to enter a disc device code. Again, the SMD disc is used.
- The SDT can be mounted on any configured tape drive. Supply its channel and subaddress.
- You can choose cold or warm start depending on whether you are relocating the SMD or changing its size. (If you use a cold start, be sure that all files have been backed up to tape before you start.)
- If you have added J.TINIT to the key files on the user SDT, it will be activated by J.TSM before J.TSM activates OPCOM.

J.TINIT checks for a file named LOGONFLE, which is built via the EDITOR as documented in Chapter 5, Volume 1. If LOGONFLE does not exist and J.TINIT is part of the central system, J.TINIT uses default terminal parameters described at the beginning of this volume (Chapter 2) and prints:

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

If LOGONFLE has been built, J.TINIT initializes terminals according to the hardware parameters specified in LOGONFLE and indicates that setup is complete as shown above.

5. USING THE ONLINE RESTART COMMAND

The TSM RESTART command is used to:

- o one-shot test a new system before creating a SDT
- o restart the current default operating system after a test
- o establish a new default system
- o return to the last system booted from a SDT as the default system

Figures 5-1 and 5-2 illustrate the backup (or default) system concept implemented in MPX-32.

5.1 Precautions

Online RESTART is a powerful command. Before using it:

- o Use the TSM SIGNAL command to notify all terminal users to stop interactive and batch activity and log off.
- o Use the OPCOM LIST command to see if any users remain logged on and abort them by task number via the OPCOM ABORT command.
- o Check batchstream activity. It should wind down as users log off. (If any batch I/O is in process, it may be lost because M.SID and M.SOD directories may be reinitialized.)
- o Independent tasks are killed during the restart process. All pending I/O will be lost. Independent tasks must be re-established or reactivated under the new system.
- o The RESTART processor was not designed to accomplish the level of scratchpad initialization as can be achieved through the IPL procedure. Therefore, system images whose configurations differ sufficiently to require a change in scratchpad should be brought up using the IPL procedure.

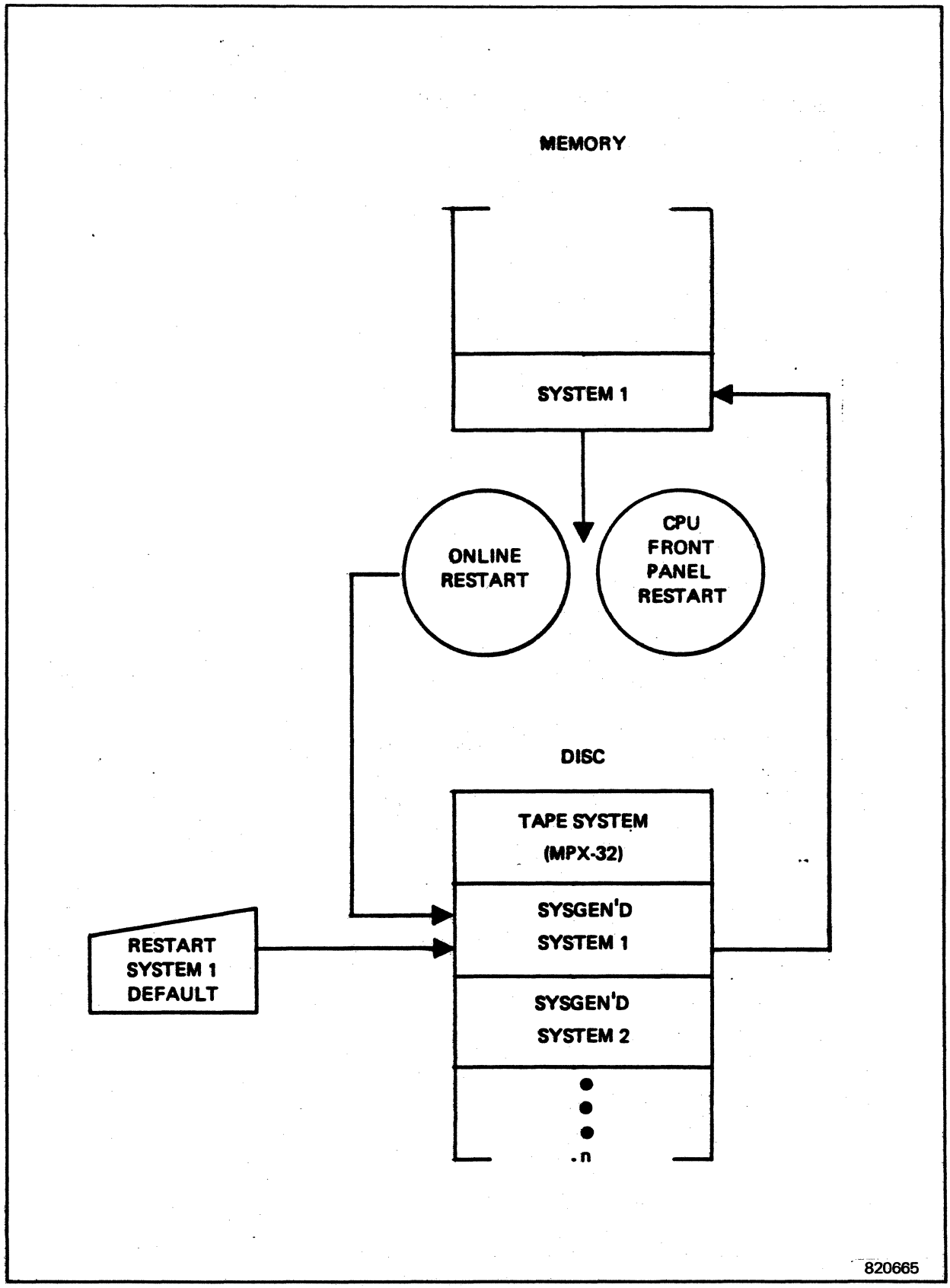

Figure 5-1

Establishing a New Default System

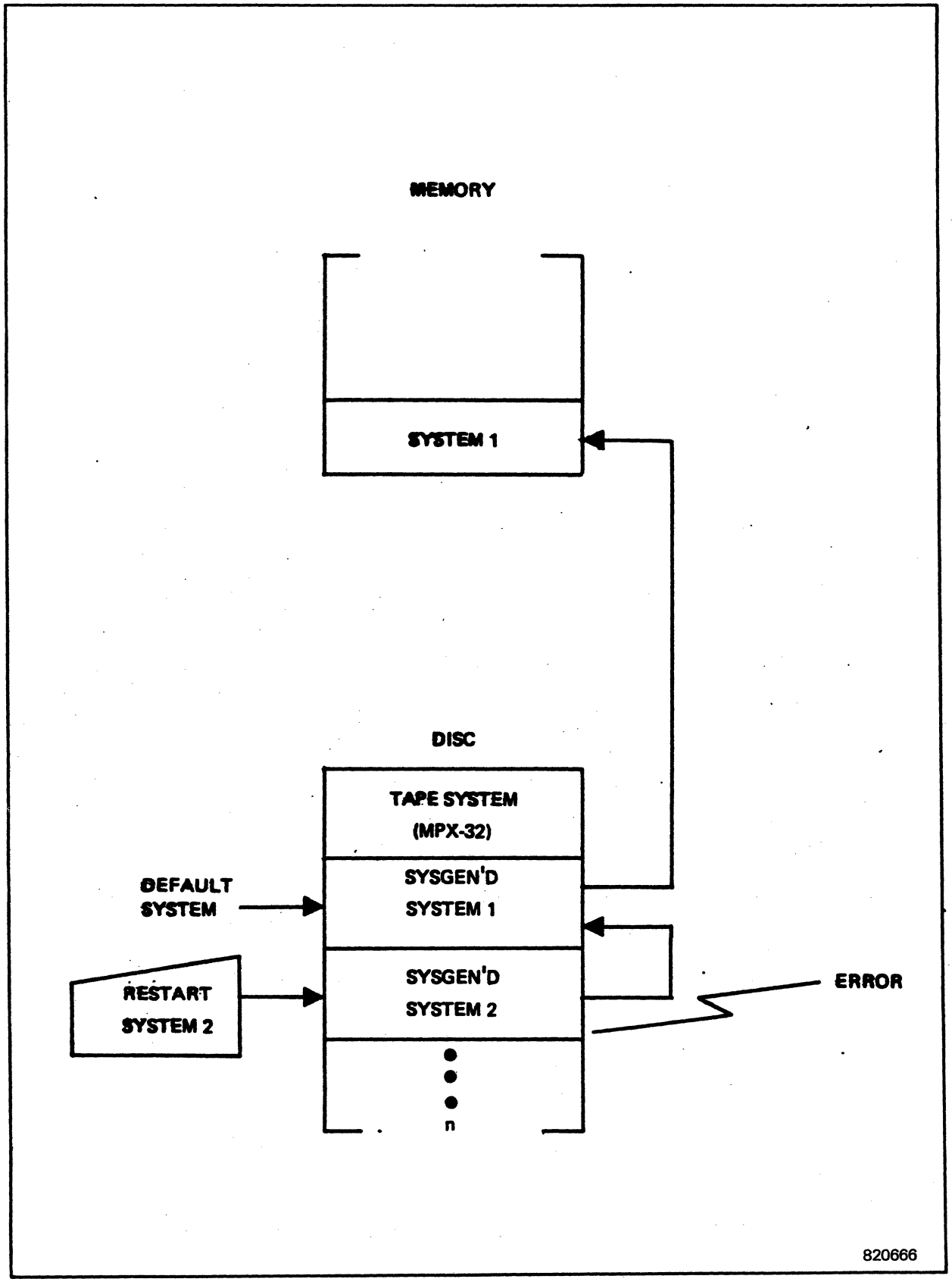

Figure 5-2

Automatic Return to a Default System

5.2 RESTART Syntax

RESTART [filename / MPX-32] [,DEFAULT]

where:

If nothing is specified, RESTART reloads the current default system as the resident operating system.

filename — is the name of the file containing the operating system to be restarted. This is the name supplied with the SYSTEM directive when a configuration of the MPX-32 system is SYSGEN'd.

MPX-32 — is used to restart the resident operating system that was booted most recently from a SDT, whether through a cold start or a warm start process.

DEFAULT — is a key word used to establish the specified system as the default system for subsequent RESTART commands as well as for restart from the CPU front panel (see the next section).

6. RECOVERING THE SYSTEM

If the operating system halts, a fresh copy of the disc file containing the operating system image can usually be restored into memory via the CPU front panel. This is a restart operation and goes through the restart cycle illustrated in the previous section.

The user can warm start from a SDT; however, warm start from tape is usually required only when the image of the system stored on the disc is damaged.

Cold start after a crash is required only if the System Master Directory has been irretrievably damaged.

6.1 Recovery from Disc at the CPU Front Panel

To restart the default MPX-32 system from disc, enter the address of the SMD disc at the front panel and depress the IPL button. RESTART logic is always located on the SMD disc and will direct processing to the most recent default system file regardless of its disc location.

6.2 Recovery Using a System Distribution Tape (SDT)

Warm start from a SDT is described in Chapter 2. A sample warm start which rebuilds the resident operating system only (no other key load modules are damaged) is illustrated below. If a key load module has been damaged (FILEMGR, J.TSM, J.INIT), resume operation from the SDT rather than abort when SYSBUILD prompts for abort or resume.

CPU Front Panel	SYSTEM RESET KEYBOARD 1000 IPL	IOP Console	RST IPL=1000 IPL=7EF0	 (if magnetic tape) (if floppy disc)

```
OPCOM
Console     COLD START OR WARM START (C/W)? W
            TASK SYSBUILD,02000001 MOUNT SCRA VOL ON M91000
            DEV,R,A,H? A

            SYSTEMS MPX-32
            INITIALIZATION COMPLETE
            TERMINAL SETUP COMPLETE
            ENTER DATE AND TIME
            ??
```

7. SYSTEM GENERATION (SYSGEN)

System Generation for an MPX-32 system involves supplying a set of configuration directives to the SYSGEN utility. The result of this step is the creation of a permanent file containing the installation specific MPX-32 system in memory image absolute format.

7.1 General Description

SYSGEN is a privileged MPX-32 system utility that operates within the framework of a standard MPX-32 system and can be executed either in batch or interactively. The system on which SYSGEN is executed must have enough memory free to hold the generated system and SYSGEN itself. SYSGEN requires 16KW.

The resources that SYSGEN requires in order to operate are as follows:

Directives
System object modules
File to contain the system resident image
System symbol table file

System object modules include interrupt and trap processors (H.IP00, H.IPIT, etc.), modules that form the MPX-32 nucleus (H.EXEC, H.ALOC, device handlers, etc.), and resident system tasks such as the Swapper (J.SWAPR) and the System Debugger (H.DBUG). These are provided as files to be restored from the master SDT after a starter system has been installed. The file naming convention for SYSTEMS-provided SDT files that are designed to be part of the resident system is OH.module.

User object modules for interrupt handlers (see the PRIORITY directive), resident system tasks (see the PROGRAM directive), and user-callable modules (M.CALL module, entry point) defined via the MODULE directive must also be included on the object file for SYSGEN.

On the master SDT, SYSTEMS provides a task for object code concatentation named COMPRESS that selects all required system module files for SYSGEN into the SYSGEN object input file. The user can modify the COMPRESS input file to select any files containing modules, interrupt handlers, resident tasks, or device handlers he wants to configure in the resident operating system. See Section 3.1.

Any handlers or tasks that are SYSGEN'd must be Assembler object modules that conform to a basic structure that begins with a Halfword Address Table (HAT), ends with an initialization entry point, and uses the following system macros: M.EIR, M.XIR, M.MODT, and M.SVCT. A task that is structured to be incorporated in the resident operating system (via the SYSGEN PROGRAM directive) is illustrated in Section 7.7.

The directives must be in card image format and may be supplied via batch from magnetic tape, disc, card reader or interactively. (See Section 7.5.) The file for the resident system image and the system symbol table file are permanent disc files created by SYSGEN.

System tables are constructed and linked to the resident system modules, handlers, and user-supplied resident modules and handlers as specified via SYSGEN directives. A resident system image is formed and subsequently written to the dynamically acquired

disc file specified via the SYSTEM directive. Concurrent with this process, a listing of directives is built and a load map of the system is generated. The load map can be saved on a system symbol table file specified by the user with the SYMTAB directive and used subsequently in patching the system.

The load map should be checked for errors. Some errors (such as NFOUND) may impact system usage (if a routine is missing, services or programs needing the routine will not be able to work). The user should decide if the error will impact the systsem before using the new system.

For further description of file assignments, see Section 7.2.

7.2 SYSGEN Files and File Assignments

All assignments for SYSGEN are provided by SYSGEN itself as defaults. They are shown in the chart which follows. The user can override the defaults with batch or TSM ASSIGN statements where shown.

All SYSGEN directives may be entered from either cards, magnetic tape, disc files, or terminals.

Table 7-1

SYSGEN File Assignments

Input/Output Description	Logical File Code	Default and Optional Assignments	How Built (Previous Processor)	How Specified	Comment
SYSGEN Directives	DIR	DIR=SYC	Work file built using EDITOR. Permanent file built using EDITOR or MEDIA. Cards or other device media, e.g., magnetic tape permanent file.	EDT > BATCH EDT > BATCH jobfile or ??BATCH { D,devmnc / F,jobfile }	SYC automatically defaults to the user's terminal if running interactively. Directives are issued one at a time. If magnetic tape, can physically include directives, SYSGEN OBJ (resident modules) and MPX-32 modules.
Directives used, Load Map, and Error Listing for Target System	LOF	LOF=SLO,1000 Override Option: ASSIGN1 LOF=filename ASSIGN3 LOF=devmnc ASSIGN4 LOF=UT	By SYSGEN	Automatic unless options are used.	
System Resident Module File	OBJ	ASSIGN1 OBJ=OH.7X Override Option: ASSIGN1 OBJ=filename	Built by COMPRESS User can input disc copy of system modules.	Automatic unless disc file option is used.	
System Symbol Table File	N/A	N/A	Output by SYSGEN on file specified with SYMTAB directive.	SYSGEN SYMTAB directive.	SYMTAB's must be on a disc file. Symbolic Table of Resident files are produced. Unique names are given to duplicate handlers.
Resident System Image File (SYSGEN'd) Target System)	N/A	N/A	By SYSGEN on file specified with SYSTEM directive.	SYSGEN SYSTEM directive.	

7.3 Options

None.

7.4 Using SYSGEN Directives

SYSGEN directives consist of three major directive types: SECTION - those beginning with //; SUB-SECTION - those beginning with /; and KEYWORD - those beginning with a predetermined keyword.

In general, the order or presentation of SYSGEN directives is not critical within a directive subsection. However, in the several instances where such order is critical, the individual directive discussion in Section 7.6 clarifies the proper order.

7.5 Accessing SYSGEN

To access SYSGEN as part of a batch job, create a job file using the EDITOR, punch cards, or other media. The job file can be read to SYC and the job activated in several ways:

from the OPCOM console:

"<ATTENTION>"

```
??BATCH  {F,jobfile}
         {D,devmnc}
```

from the OPCOM program:

```
TSM>OPCOM
??BATCH  {F,jobfile}
         {D,devmnc}
```

from the EDITOR:

```
TSM > EDIT
        .
        .
        .
EDT > BATCH  jobfile
```

If the job file is the current EDITOR work file, issue just the BATCH command.

To activate SYSGEN and run online, use the TSM ASSIGN commands to make SYSGEN assignments, if needed, then proceed to issue SYSGEN directives.

```
TSM> SYSGEN
SYS>//HARDWARE
SYS>/PARAMETERS
SYS>etc.
```

7.6 SYSGEN Directives

SYSGEN directives all begin in byte 1 of the record and contain no embedded blanks. Numeric values are represented by decimal numbers unless otherwise specified.

All SYSGEN directives are required unless specifically described as optional. Directives that begin with two slashes indicate the beginning of major sections (//HARDWARE and //SOFTWARE, plus //END). Directives that break out sections within the major sections begin with one slash; directives that are subsections within a section have no slash.

Directive	Function
TITLE	The TITLE directive permits identifying information to be printed on the listed output file.
//HARDWARE	The //HARDWARE directive delimits the section of SYSGEN directives pertaining to the user's hardware configuration.
/PARAMETERS	The /PARAMETERS directive designates the parameters subsection of the //HARDWARE section.
MACHINE	The MACHINE directive indicates what type of computer the system is being configured for.
IPU	The IPU directive specifies an Internal Processing Unit will be configured on the target system.
MEMONLY	The MEMONLY directive indicates the configured system will be a memory-only system.
/MEMORY	The /MEMORY directive designates the MEMORY subsection of the //HARDWARE section.
TYPE	The TYPE directives specify the memory configuration of the target system.
/INTERRUPTS	The /INTERRUPTS directive designates the interrupt directives subsection.
PRIORITY	The PRIORITY directives specify the interrupt configuration and the interrupt processors to be used by the target system.
/TRAPS	The /TRAPS directive delimits the traps description subsection.
PROGRAM	The PROGRAM directives specify the names of configured trap handlers and resident system tasks.
/CHANNELS	The /CHANNELS directive delimits the controller and device directives subsection.
CONTROLLER	Each CONTROLLER directive represents one hardware channel to be configured to the generated MPX-32. Non-present channels may be configured if desired.

DEVICE — The DEVICE directives define configured hardware I/O devices. Non-present devices may be configured if desired. Non-configured discs, however, must be specified as offline.

/SYSDEVS — The /SYSDEVS directive delimits the system device definition section of directives.

SID — The SID directive specifies a system input device (SID). Used as the default device in related OPCOM commands.

LOD — The LOD directive specifies a system listed output device. Used as the default device in related OPCOM commands.

POD — The POD directive specifies a system punched output device. Used as the default device in related OPCOM commands.

SWP — The SWP directive specifies the system swapping disc device.

//SOFTWARE — The //SOFTWARE directive indicates the beginning of the software section of directives.

/PARAMETERS — The /PARAMETERS directive indicates the beginning of the /PARAMETERS subsection of the //SOFTWARE section.

DISP — The DISP directive determines the number of entries in the MPX-32 dispatch queue. One entry is required for each concurrently operating program. Each entry requires forty-two words of resident image storage. A minimum of four entries is required.

POOL — The POOL directive specifies the size of the memory pool to reserve.

MTIM — The MTIM directive provides the number of Real Time Clock interrupts per second.

NTIM — The NTIM directive provides the number of Real Time Clock interrupts per time unit.

ITIM — The ITIM directive provides the expiration time interval of the RTOM interval timer.

SYSTEM — The SYSTEM directive provides the name of the file to be used as storage for the generated MPX-32 resident image.

ITLB — The ITLB directives provide for generation of an Indirectly Connected Task Linkage Blocks.

SYMTAB — The SYMTAB directive provides the name of the file to be used as storage for the system symbol table. A listing of the symbolic table, resident and unique names given to duplicate handlers can be produced by specifying the option SYM with /O directive in the M.PATCH file.

PASSWORD — The PASSWORD directive specifies the password to associate with SYSGEN-created permanent disc files containing system programs. If this directive is included, the files created by SYSGEN are read-only files with the specified password. The spooled input directory (M.SID and spooled output directory (M.SOD) are also created as read-only files with the specified password.

TQFULL	The TQFULL directive specifies the largest time quantum that a single user time distribution task acquires prior to being relinked to the bottom of the priority list at its base execution priority.
TQMIN	The TQMIN directive specifies the smallest time quantum that a single user task acquires before preemption by a higher priority user time distribution task.
BATCHPRI	The BATCHPRI directive provides the execution base priority for batch jobs.
TERMPRI	The TERMPRI directive specifies the execution priority level for all tasks activated in the interactive terminal environment. Default: 60.
PATCH	The PATCH directive specifies a patch area to append to the MPX-32 resident image.
MODE	The MODE directive is used to request special system operations.
SVC	The SVC directive is used to optionally increase the size of the SVC table.
FLTSIZE	The FLTSIZE directive is used to increase the size of the File Lock Table.
RMTSIZE	The RMTSIZE directive is used to increase the size of the Resourcemark Table.
ACTIVATE	The ACTIVATE directive specifies the load module name of programs to be activated after the system has been booted.
TRACE	The TRACE directive allows initialization of the System Trace flag word C.TRACE.
DEBUGTLC	The DEBUGTLC directive allows specification of the console address for the System Debugger's stand-alone I/O.
/MODULES	The /MODULES directive defines the begining of the /MODULES subsection of the //SOFTWARE section.
MODULE	The user MODULE directive 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	The /OVERRIDE directive defines the beginning of the /OVERRIDE subsection of the //SOFTWARE section.
SYSMOD	The SYSMOD directive allows the user to replace a system module with another module.
/PARTITION	The /PARTITION directive informs SYSGEN that memory partitioning and/or global common is requested and that partition definition directives follow.

NAME	The NAME directives define DATAPOOL and/or Global Common memory partitions.
/TABLES	The /TABLES directive defines the beginning of the tables subsection of the //SOFTWARE section.
SHARE	The SHARE directive specifies the number of entries in the shared memory table.
JOBS	The JOBS directive specifies the number of entries in the job table; this determines the maximum number of batch jobs which may be active concurrently.
TIMER	The TIMER directive specifies the number of timer entries to be generated in the MPX-32 resident image.
/FILES	The /FILES directive identifies the system files section of the //SOFTWARE directives.
SMD	The SMD directive defines the System Master Directory (SMD) disc.
SYCSIZE	The SYC directive specifies the initial size of the spooled input file allocated for each batch job. A job's spooled input file is expanded as required. However, an initial size should be specified that accommodates most jobs to preclude frequent file expansion.
SGOSIZE	The SGO directive specifies the default size of the SGO file to allocate for each batch job.
//END	The //END directive is required as the last SYSGEN directive.

7.6.1 TITLE Directive

The TITLE directive provides a mechanism for attaching identifying information to the SYSGEN listed output. If supplied, this directive must be the first in the directive stream.

Syntax:

TITLE = data

where:

data are from 1-65 ASCII characters

7.6.2 //HARDWARE Directive

The //HARDWARE directive delimits the section of SYSGEN directives pertaining to a given hardware configuration. Required.

Syntax:

//HARDWARE

7.6.3 /PARAMETERS Directive

The /PARAMETERS directive designates the parameters subsection of the //HARDWARE section. Required. It must be the first subsection of the //HARDWARE section.

Syntax:

/PARAMETERS

7.6.3.1 MACHINE Directive

The MACHINE directive indicates to the SYSGEN task what type of computer the resultant system is being configured for. Required. It must be the first directive in the /PARAMETERS subsection.

Syntax:

MACHINE = type

Where:

type is the machine type 327X, 3227, or 3287.

7.6.3.2 IPU Directive

The IPU directive specifies that an Internal Processing Unit will be configured in the target system. See MPX-32 Reference Manual, Volume 1, Section 2.3 for a description of the IPU.

Syntax:

IPU

Note: This directive causes the modules H.IPU and H.CPU to be loaded.

7.6.3.3 MEMONLY Directive

The MEMONLY directive specifies the configured system will be a memory-only system.

Syntax:

MEMONLY

7.6.4 /MEMORY Directive

The /MEMORY directive designates the /MEMORY subsection of the //HARDWARE section. Required in /PARAMETERS section.

Syntax:

/MEMORY

7.6.4.1 TYPE Directive

The TYPE directives specify the memory configurations of the target MPX-32 system. They are required in the /MEMORY subsection. Ordering of these directives by starting page number is essential.

Syntax:

SIZE = nn, TYPE = c, CLASS = x

where:

nn is the size of the memory being described by this directive expressed in decimal number of 8K (32/7x) or 2K (CONCEPT/32) map blocks.

c memory type 'E', 'H', 'S', 'N':

- E is the first 128KW in the system
- H is memory beyond 128KW with 600 ns access time
- S is memory beyond 128KW with 900 ns access time
- N is absent memory

x memory class 'C' or 'S':

- C is core memory
- S is semiconductor memory

7.6.5 /INTERRUPTS Directive

The /INTERRUPTS directive designates the interrupt directives subsection. Required within the //HARDWARE section.

Syntax:

/INTERRUPTS

7.6.5.1 PRIORITY Directive

The PRIORITY directives specify the interrupt configuration and interrupt processors to use for the target system. Interrupt processors provided by SYSTEMS are: Attention (H.IP13), Call Monitor (H.IP27), Real Time Clock (H.IP28), and CPU Scheduler (H.IPIT) (which is directly connected to the lowest interrupt priority level in the system, i.e., 5F on the CONCEPT/32 or 7F on a 32/7X).

In addition, 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 priority 77 if a Scientific Accelerator is not configured in the system. If a Scientific Accelerator is present, a priority of 3F should be used for IPU accounting.

Syntax:

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

where:

intlev	is the two hexadecimal digit interrupt level (lowest levels are 5F on the CONCEPT/32 or 7F on the 32/7X).
channel	is the two hexadecimal digit RTOM board or IOP RTOM function channel address.
subaddress	is the two hexadecimal digit RTOM board or IOP RTOM function subaddress for this interrupt level which is the 1's compliment of the RTOM relative physical priority.
name	name of program (one to eight characters) located on lfc OBJ. If program is to be directly connected to this interrupt level and SYSGEN'd with the resident operating system, the name is supplied. If tasks are to be indirectly connected to the interrupt level, no program name is supplied; an Indirectly connected Task Linkage Block (ITLB) is defined via the ITLB directive. (See Section 7.6.9.6.)
INTV	is used to indicate that the level is an RTOM interval timer. This results in a device entry being built in the scratchpad whose address is equal to the interrupt priority level.

7.6.5.2 /TRAPS Directive

The /TRAPS directive delimits the trap descriptor subsection. Required.

Syntax:

/TRAPS

7.6.5.3 PROGRAM Directives

The PROGRAM directives specify the program names of the trap processors to be configured on the resident system. These include Power Fail-Safe/Autostart (H.IP00), System Override (H.IP01) and other trap processors provided by SYSTEMS. (See Volume 1, Figure 1-2.) The PROGRAM directive can also be used to configure user tasks in the resident system. A sample user task and restrictions are provided in Section 7.7. The only special requirement is a handbuilt TSA as illustrated in the example.

Syntax:

PROGRAM = (name1,. . . name7)

where:

name1...name7	are 1-8 character ASCII program names. The object of these programs must have been compressed and located on lfc OBJ. A maximum of 7 names separated by commas can be entered per directive.

7.6.6 /CHANNELS Directive

The /CHANNELS directive delimits the controller and device directives subsection of //HARDWARE. Required.

Syntax:

/CHANNELS

7.6.6.1 CONTROLLER Directive

Each CONTROLLER directive represents one hardware channel to be configured in the generated MPX-32 system. Required for disc-based MPX-32 systems. The handler key word processing permits specification of reentrancy and of 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 UDT specification applies until another HANDLER keyword or CONTROLLER statement is processed.

Note: nonpresent channels may be configured if desired.

The null device, device type code NU, is required to be included in every configuration. See example in Section 7.7.

Syntax:

$$\text{CONTROLLER} = \text{ttcc, PRIORITY} = \text{intlev, CLASS} = \text{class, HANDLER} = \left\{ \begin{matrix} \text{name} \\ (\text{name}\left[,\begin{matrix}\text{I}\\\text{S}\end{matrix}\right]\left[,\text{C}\right]) \end{matrix} \right\}$$

$$[,\text{MUX} = \text{type}]\ [,\text{SUBCH=aa}]$$

where:

tt	is the two ASCII character device code (refer to Table 7-2).
cc	is the two hexadecimal digit channel number.
intlev	is the two hexadecimal digit interrupt level.
class	is the device class, as follows: 0 = TLC Line Printer 1 = TLC Card Reader 2 = TLC Teletype D = 16mb Addressable E Class E = All Others F = Extended I/O
name	is the one- to eight-character handler name.
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
type	is used if a multiplex controller is being configured: GPMC - General Purpose Multiplex Controller XIO - Extended I/O (FMS,addr) - FMS interface, where addr is the two digit hexadecimal control channel address. Note: Not a standard product. IOP - Input/Output Processor QGPMC - Old General Purpose Multiplex Controller
aa	is the IOP subchannel which the controller is connected to and will be used to verify proper device address specifications on subsequent device directives (i.e., the subchannel should match the first device address digit).

Notes:

Extended I/O handlers default to system reentrant handlers.

GPMC device handlers default to system reentrant handlers.

QGPMC device handlers default to interrupt level reentrant handlers. A CDT is generated each time a HANDLER key word appears, except on the first DEVICE statement following a CONTROLLER statement with a HANDLER specification.

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

Dev Type Code	Device	Device Description
00	CT	Operator Console (Not Assignable)
01	DC	Any Disc Unit
02	DM	Any Moving Head Disc
03	DF	Any Fixed Head Disc
04	MT	Any Magnetic Tape Unit
05	M9	Any 9-Track Magnetic Tape Unit
06	M7	Any 7-Track Magnetic Tape Unit
07	CD	Any Card Reader-Punch
08	CR	Any Card Reader
09	CP	Any Card Punch
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 Disc
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)

Notes

(1) Standard I/O handlers for MPX-32 are referenced in Table 7-4.

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

(3) Mnemonic 'NU' is the null device.

(4) Mnemonics U0-U9 are available for user-defined applications.

Table 7-2

MPX-32 Device Codes

7.6.6.2 DEVICE Directive

DEVICE directives define the configured hardware I/O devices. Required. The handler key word processing permits specification of reentrancy and of 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 UDT specification applies until another HANDLER key word or CONTROLLER statement is processed. Note: nonpresent device may be configured if desired. Then, when the device is added, a warm start with the device marked on-line will allow it to be used.

The null device, device type code NU, is required to be included in every configuration. See example in Section 7.7.

Syntax:

$$\text{DEVICE} = \begin{Bmatrix} aa \\ (aa,n,inc) \end{Bmatrix} \left\{ \left[,\text{DISC} = \begin{Bmatrix} devcode \\ (devcode[,D]\ ) \end{Bmatrix} \right] \right. [\text{DPTOV=time}][,\text{SHR}][,\text{DTC=tt}]$$

$$[,\text{LINSIZ} = x] \quad [,\text{PAGE} = y][,\text{SPOOL} = (code,code,...)]$$

$$\left[,\text{HANDLER} = \begin{Bmatrix} name \\ (name[,I][,S][,C]) \end{Bmatrix} \right] \quad [,\text{PHYSA} = ccaa]\ [,\text{OFF}]\ [,\text{IOO=mode}]$$

where:

aa	is the two hexadecimal digit device subaddress.
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.
devcode	is the four-character device code for disc storage devices. It is not required for devices other than disc. (Refer to Table 7-3)
D	is an optional parameter specifying a dual port disc.
time	is the number of timer units a CPU will wait before taking away a dual ported disc from an opposing CPU. Default: 2.
SHR	is the optional designation as a shared device. Note: all disc devices are shared.
tt	is the optional two character device mnemonic from Table 7-2. If not specified, the device mnemonic specified on the associated CONTROLLER directive is used.
x	is an optional parameter for TSM devices which specifies characters per line. The valid range is 7-255, inclusively.

y — is an optional parameter for TSM devices which specifies lines per screen. Zero can be used for a hard-copy device.

code — is the optional specification that the device is available for automatic selection as the destination device for spooled printed (SLO) and punched (SBO) output. The specification consists of a set of two-character codes separated by commas. The codes "BL", "BB", "RL" and "RB" indicate that the device is to be used for output of batch SLO, batch SBO, real-time SLO, and real-time SBO, respectively.

name — is the one- to eight-character handler name. If not specified, the handler name on the associated CONTROLLER directive is used.

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.

PHYSA= — specifies the physical (bus) channel address and device subaddress for devices; mandatory for TLC's. Used when logical channel and subaddress do not match the physical channel and subaddress.

cc — is the two digit hexadecimal channel address.

OFF — specifies that the device(s) described by this directive are to be SYSGEN'd in OFFLINE state.

mode — is used by the IOP and GPMC to indicate either the I/O queue entries are to be linked from the UDT (IOO=DEV) or from the CDT (IOQ=CONT). Default: IOQ=CONT.

Notes:

Extended I/O handlers default to system reentrant handlers.

GPMC device handlers default to system level reentrant handlers.

QGPMC device handlers default to interrupt level reentrant handlers.

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

Disc	Disc Code
4MB Fixed Head Disc - Class E Device	FE004
10MB Cartridge Module Disc - Class E Device	CE010
40MB Moving Head Disc - Class E Device	ME040
80MB Moving Head Disc - Class E Device	ME080
300MB Moving Head Disc - Class E Device	ME300
5MB Fixed Head Disc - Class E Device	FE005
1.2MB Floppy Disc - Class F Device	FL001
40MB Moving Head Disc - Class F Device	MH040
80MB Moving Head Disc - Class F Device	MH080
300MB Moving Head Disc - Class F Device	MH300
5MB Fixed Head Disc - Class F Device	FH005
32MB Cartridge Module Disc - Class F Device	CD032

Table 7-3 Disc Device Codes

Handler Module Name	Description
H.DF01	Fixed Head Disc
H.DM01	Moving Head Disc
H.CD00	Cartridge Disc Handler
H.DP01	Extended I/O Moving Head Disc Phase 1
H.DP02	Extended I/O Moving Head Disc Phase 2
H.ASMP	GPMC Async Comm. for Terminals
H.CR00	Card Reader
H.CT00	TLC OPCOM Console
H.LP00	Line Printer
H.CPMP	GPMC Card Reader-Punch
H.MT00	Magnetic Tape
H.XMT	Extended I/O Magnetic Tape
H.TY10	ADS Terminal Interface
H.MUX0	GPMC Multiplexer
H.EXIO	Extended I/O Multiplexer
H.BSMP	Bisync GPMC
H.PTMP	Paper Tape GPMC
H.ADIO	Analog Digital Handler
H.SLMP	SLIM Handler
H.FLIOP	IOP Floppy Disc
H.A8IOP	IOP 8-Line Asynchronous Handler
H.F8IOP	Full Duplex 8-Line Async Handler
H.CTIOP	IOP Console

Table 7-4 Device Handlers

7.6.7 /SYSDEVS Directive

The /SYSDEVS directive delimits the system device definitions. Required for disc-based MPX-32 systems. Not required for memory-only MPX-32 systems.

Syntax:

/SYSDEVS

7.6.7.1 SID Directive

The SID directive specifies a system input device (SID) to use as the default device in related OPCOM commands.

Syntax:

SID = devmnc [,DENSITY = density][,PARITY = parity]

where:

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

density — H for high density
L for low density
Used only for 7-track magnetic tape.

parity — E for even parity
O for odd parity
Used only for 7-track magnetic tape.

7.6.7.2 LOD Directive

The LOD directive specifies a system listed output device to use as the default device in related OPCOM commands. Required in /SYSDEVS if a line printer is configured.

Syntax:

LOD = devmnc [,IBP]

where:

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

IBP — used to inhibit banner page produced on System Listed Output device.

7.6.7.3 POD Directive

The POD directive specifies a system punched output device to use as the default device in related OPCOM commands. Required if a card punch is configured.

Syntax:

POD = devmnc

where:

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

7.6.7.4 SWP Directive

The SWP directive specifies the disc device from which the temporary swap files for task outswap will be acquired. This directive is optional and if not supplied the swap space will be acquired from the SMD disc.

Syntax:

SWP = devmnc

where:

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

7.6.8 //SOFTWARE Directive

The //SOFTWARE directive indicates the beginning of the software configuration section of directives. Required.

Syntax:

//SOFTWARE

7.6.9 /PARAMETERS Directive

The /PARAMETERS directive indicates the beginning of the /PARAMETERS subsection of the //SOFTWARE directive section. Required.

Syntax:

/PARAMETERS

7.6.9.1 DISP Directive

The DISP directive determines the number of entries in the MPX-32 dispatch queue. One entry is used for each concurrently operating task. If not specified, SYSGEN defaults to 10.

Syntax:

DISP = entries

where:

entries specifies the number of entries in the dispatch queue. This value cannot exceed 255 and must be at least 4.

7.6.9.2 POOL Directive

The POOL directive specifies the size of the memory pool to be reserved at SYSGEN time. If not specified, the size of memory pool defaults to 1000 words.

Syntax:

POOL = words

where:

words is the number of words to be reserved.

7.6.9.3 NTIM Directive

The NTIM directive provides the number of Real Time Clock interrupts per time unit. If not used, SYSGEN defaults to 60 interrupts per time unit.

Syntax:

NTIM = number

where:

number is the number of clock interrupts per time unit.

7.6.9.4 MTIM Directive

The MTIM directive provides the number of Real Time Clock interrupts per second. If not used, SYSGEN defaults to 60 interrupts per second.

Syntax:

MTIM = number

where:

number is the number of Real Time Clock interrupts per second.

7.6.9.5 ITIM Directive

The ITIM directive provides the expiration time interval of the RTOM interval timer. If not specified, SYSGEN defaults to an expiration interval of 38.4 microseconds.

Syntax:

ITIM = 10^{-7} second

where:

10^{-7} second	is the decimal value to the tenth of a microsecond for the RTOM interval timer time quantum, i.e., 38.4 microseconds is represented as 384.

7.6.9.6 ITLB Directive

The ITLB directive provides for the generation of an Indirectly Connected Task Linkage Block. One ITLB directive is required for each indirectly linked task to be concurrently active in the system, and must be connected through a PRIORITY directive.

Syntax:

ITLB = intlevel

where:

intlevel	is the two hexadecimal character interrupt priority level to which the task is to be indirectly connected.

7.6.9.7 PASSWORD Directive

The PASSWORD directive specifies a password to be associated with SYSGEN-created permanent disc files. Optional.

If PASSWORD is not included, the disc files are created without restricted access.

Syntax:

PASSWORD = password

where:

password	is a one- to eight-character password. The password may not contain an equal sign, comma, left parenthesis, or right parenthesis.

7.6.9.8 SYSTEM Directive

The SYSTEM directive provides the name of the permanent file to which the generated MPX-32 system will be written. If the file does not currently exist, SYSGEN will create

the file. Note that if the PASSWORD directive is supplied, that password is also applicable to the file containing the resident system image. Required.

Syntax:

SYSTEM = sysfile

where:

sysfile — is the ASCII file name of the file to contain the resident system image generated at SYSGEN.

7.6.9.9 SYMTAB Directive

The SYMTAB directive provides the name of the permanent file to which the symbol table of the generated MPX-32 system will be written. The file, if it does not currently exist, will be created. The password specified via the PASSWORD directive will be used if supplied. Required.

Syntax:

SYMTAB = filename

where:

filename — is the ASCII file name of the system symbol table file. A listing of the symbolic table, resident and unique names given to duplicate handlers can be produced by specifying the option SYM with the /O directive in the M.PATCH file.

7.6.9.10 TQFULL Directive

The TQFULL directive specifies the largest time quantum that a time distribution task acquires prior to preemption by another task at the same priority level. If either TQFULL or TQMIN directives are supplied, then both are required. Note that TQFULL must be greater than TQMIN. If not specified, SYSGEN defaults to 1200 milliseconds.

Syntax:

TQFULL = time

where:

time — is the number of milliseconds

7.6.9.11 TQMIN Directive

The TQMIN directive specifies the minimum time quantum that a time distribution task may acquire before pre-emption by another time distribution task at a higher base priority level. If either TQMIN or TQFULL directives are supplied, then both are required. If not specified, SYSGEN defaults to 400 milliseconds.

Syntax:

TQMIN = time

where:

time — is the number of milliseconds.

7.6.9.12 BATCHPRI Directive

The BATCHPRI directive specifies the execution priority level of all batch jobs. If BATCHPRI is not used, SYSGEN defaults to priority 61 for batch jobs.

Syntax:

BATCHPRI = nn

where:

nn — is the two digit decimal time distribution priority level (55-64) to use for batch jobs.

7.6.9.13 TERMPRI Directive

The TERMPRI directive specifies the execution priority level for all tasks activated in the interactive terminal environment. If TERMPRI is not used, SYSGEN defaults to priority 60.

Syntax:

TERMPRI = nn

where:

nn — is the two digit decimal time distribution priority level (55-64) to use for interactive processing.

7.6.9.14 PATCH Directive

The PATCH directive specifies a patch area to append to the MPX-32 resident image.

Syntax:

PATCH = number

where:

number — specifies the hexadecimal number of bytes to be added to the resident MPX-32 system as a system patch area.

7.6.9.15 MODE Directive

The MODE directive is optional and is used to request the following special system operations:

Continuous Batch - Batchstream 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 which is produced by system output tasks when processing SLO files.

Uni-Directional File Allocation - Treats all requests for temporary disc space as permanent file requests, i.e., space is allocated from the high end of the disc downward.

Inhibit Tape Mount Message - Suppresses tape mount message output on the system console (this option is ignored for multivolume tapes).

Dump - Indicates a dump is to be output if an independent task aborts.

Scratchpad Locations - When set, indicates unused CPU scratchpad locations are not to be zeroed at IPL.

Syntax:

$$\text{MODE} = \left\{ \begin{array}{l} \text{SCBT} \\ \text{SIBP} \\ \text{SUFA} \\ \text{SIMM} \\ \text{DUMP} \\ \text{LSPA} \end{array} \right\}$$

where:

SCBT	sets continuous batch.
SIBP	sets inhibit banner page.
SUFA	sets uni-directional file allocation
SIMM	sets inhibit magnetic tape mount message.
DUMP	sets dump request for aborting real-time tasks.
LSPA	if set, IPL process does not zero unused scratchpad locations. Default, IPL process zeroes all unused CPU scratchpad locations.

Note:

All of the above special operations can be reset through the OPCOM MODE command once the system is running.

7.6.9.16 SVC Directive

The SVC directive is used to increase the size of the SVC table. Optional. If not specified, the SVC table size defaults to 7F.

Syntax:

SVC = num

where:

num	is a hexadecimal number between X'80' and X'FF'.

7.6.9.17 FLTSIZE Directive

The FLTSIZE directive is used to increase the size of the File Lock Table. Optional. If not specified, FLT size defaults to 200.

Syntax:

FLTSIZE=num

where:

num is a decimal number between 200 and 1,000.

7.6.9.18 RMTSIZE Directive

The RMTSIZE directive is used to increase the size of the Resourcemark Table. Optional. If not specified, RMT size defaults to 64.

Syntax:

RMTSIZE=num

where:

num is a decimal number between 64 and 1,000.

7.6.9.19 ACTIVATE Directive

The ACTIVATE directive specifies the load module names of programs to be activated by J.INIT immediately after the target system has been booted. Optional.

Syntax:

ACTIVATE=(name1,...name7)

where:

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

7.6.9.20 TRACE Directive

The TRACE directive allows initialization of the System Trace flag word C.TRACE. See Chapter 7, Volume 1 for a description of bit indicators within C.TRACE. Optional. If not specified, C.TRACE defaults to X'FFFFFFFE'.

Syntax:

TRACE = num

where:

num is a 0-8 character hexadecimal number.

7.6.9.21 DEBUGTLC Directive

The DEBUGTLC directive allows specification of the console address for the System Debugger's stand-alone I/O. Optional. If not specified, the console address defaults to X'7E'.

Syntax:

DEBUGTLC = cc

where:

cc is a two digit hexadecimal channel number.

7.6.10 /MODULES Directive

The /MODULES directive defines the beginning of the /MODULES subsection of the //SOFTWARE section. Optional.

Syntax:

/MODULES

7.6.10.1 MODULE Directive

User MODULE directives define the names of optional user module(s) to include in the MPX-32 resident image. One MODULE directive is required for each user module to be included.

Syntax:

MODULE = (name,module,entpoints)

where:

name — is the one- to eight-character ASCII module name. The name may not contain a comma, an equal sign, or a left or right parenthesis.

module — is a two-digit decimal number representing the internal identification number of the module. Module 00 through 08 are reserved for MPX-32 modules.

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 subsequent to execution. This entry point should not be included in the entpoints value supplied on the directive.

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

7.6.11 /OVERRIDE Directive

The /OVERRIDE directive defines the beginning of the /OVERRIDE subsection of the //SOFTWARE section. Optional.

Syntax:

/OVERRIDE

7.6.11.1 SYSMOD Directive

The SYSMOD directive allows the user to replace system modules with other modules. One SYSMOD directive is required for each system module to be replaced.

Syntax:

SYSMOD = name1, REPMOD= name2

where:

name1	is the one- to eight-character ASCII name of the system module to be replaced.
name2	is the one- to eight-character ASCII name of the replacement module.

7.6.12 /PARTITION Directive

The /PARTITION directive informs SYSGEN that memory partitioning and/or global common is requested and that partition definition directives follow. Optional.

Syntax:

/PARTITION

7.6.12.1 NAME Directive

Partition NAME directives are used to define DATAPOOL, Global Common, or other memory partitions. One NAME directive is required for each memory partition desired.

Syntax:

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

where:

name	is the one- to eight-character partition name.
np	is the decimal number of pages for the partition. Maximum size for 32/7x is 192 pages (96 KW). Maximum size for CONCEPT/32 is 304 pages (152 KW).

sp — is the logical hexadecimal starting protection granule for the partition.

pm — is the physical decimal starting physical 8K (32/7x) or 2K (CONCEPT/32) map block number.

Note: The user must be sure not to specify values that will conflict with placement of the operating system.

7.6.13 /TABLES Directive

The /TABLES directive defines the beginning of the tables subsection of the //SOFTWARE directives. Required.

Syntax:

/TABLES

7.6.13.1 JOBS Directive

The JOBS directive specifies the number of entries in the job table. This defines the maximum number of batch jobs which may be active concurrently. If not supplied, the batch job table is defaulted to one (1) entry.

Syntax:

JOBS = number

where:

number — is the number of entries in the job table.

7.6.13.2 SHARE Directive

The SHARE directive specifies the number of entries in the shared memory table. Each entry defines a shared memory area, i.e., CSECT, Global Common, or DATAPOOL. Optional. If not specified, the number of entries defaults to 0.

Syntax:

SHARE = number

where:

number — is the number of entries in the shared memory table. On a 32/7x, each entry requires 16 words of memory. On a CONCEPT/32, each entry requires 48 words of memory. It must be sufficient to define all static and dynamic partitions.

7.6.13.3 TIMER Directive

The TIMER directive specifies the number of timer table entries to be generated in the MPX-32 resident image.

Syntax:

TIMER = number

where:

number is the number of timer table entries to be generated.

7.6.14 /FILES Directive

The /FILES directive identifies the System Files section of the //SOFTWARE directives. Required for disc-based MPX-32 systems. Not required for memory-only MPX-32 systems.

Syntax:

/FILES

7.6.14.1 SMD Directive

The SMD directive defines the System Master Directory (SMD) device and number of entries (default starter system size is 2000). The SMD disc always contains the SMD and the resident system image that was last booted from a SDT.

Warning: Whenever the SMD size of an existing system is changed, a cold start must be performed.

Syntax:

SMD = devmnc, ENTRIES = ents

where:

devmnc is the two character device code of the SMD disc (see Table 7-2), the two hexadecimal digit channel number, and the two hexadecimal digit device subaddress.

ents specifies the maximum number (decimal) of permanent files which may be created on the system. In order to allocate a sufficient number of SMD entries, SYSGEN doubles the "ents" specification and adds 1 word to give an odd result.

* RTM users should note that there is no longer an SPS device. The SMD and the resident system image booted from tape are always located on the disc specified with the SMD directive.

7.6.14.2 SYCSIZE Directive

The SYCSIZE directive specifies the initial size of the spooled input file (SYC) allocated for each batch job. A job's spooled input file is expanded as required. However, an initial size should be specified to accommodate most jobs to preclude frequent file expansion. If this directive is not supplied, the SYC size defaults to 20 blocks.

Syntax:

SYCSIZE = blocks

where:

blocks	is the number of 192-word blocks of disc space to be allocated initially for each SYC file.

7.6.14.3 SGOSIZE Directive

The SGOSIZE directive specifies the default size of the System General Output (SGO) file to be allocated for each batch job. If this directive is not supplied, the SGO size defaults to 20 blocks.

Syntax:

SGOSIZE = blocks

where:

blocks	is the default number of 192-word blocks of disc space to be allocated for each SGO file.

7.6.15 //END Directive

This directive is required as the last SYSGEN directive.

Syntax:

//END

7.7 Examples

7.7.1 SYSGEN Directive Example - 32/7X

```
TITLE=32/7X  DUAL 80 MB DISCS  ADS AND ALIM
//HARDWARE
/PARAMETER
MACHINE=327X
/MEMORY
SIZE=16,TYPE=E,CLASS=S
SIZE=8,TYPE=S,CLASS=S
/CHANNELS
CONTROLLER=DM08,PRIORITY=16,CLASS=E,HANDLER=H.DM01
DEVICE=(00,2,1),DISC=ME080,DTC=DM
CONTROLLER=M910,PRIORITY=18,CLASS=E,HANDLER=H.MT00
DEVICE=(00,2,1),DTC=M9
CONTROLLER=TY20,PRIORITY=1A,CLASS=E,MPX=GPMC,HANDLER=H.MUX0
DEVICE=(00,16),DTC=TY,LINSIZ=80,PAGE=24,HANDLER=H.ASMP
CONTROLLER=TY60,PRIORITY=1E,CLASS=E,HANDLER=H.TY10
DEVICE=(00,8),DTC=TY,LINSIZ=80,PAGE=24
CONTROLLER=LP7A,PRIORITY=21,CLASS=0,HANDLER=H.LP00
DEVICE=00,DTC=LP,SPOOL=(BL,RL),PHYSA=7802
CONTROLLER=CR78,PRIORITY=20,CLASS=1,HANDLER=H.CR00
DEVICE=00,DTC=CR,PHYSA=7800
CONTROLLER=CT7E,PRIORITY=23,CLASS=2,HANDLER=H.CT00
DEVICE=00,DTC=CT,PHYSA=7801,LINSIZ=132,PAGE=24
CONTROLLER=NU00
DEVICE=00,DTC=NU,SHR,SPOOL=(BB,RB)
/INTERRUPTS
PRIORITY=13,RTOM=(79,1C),PROGRAM=H.IP13
PRIORITY=27,RTOM=(79,18),PROGRAM=H.IP27
PRIORITY=28,RTOM=(79,16),PROGRAM=H.IPCL
PRIORITY=2C,RTOM=(79,13)
PRIORITY=7F,RTOM=(79,04),PROGRAM=H.IPIT,INTV
/TRAPS
PROGRAM=(H.IP00,H.IP01,H.IP02,H.IP03,H.IP04,H.IP05)
PROGRAM=(H.IP06,H.IP07,H.IP08,H.IP09,H.IP0E,H.IP0F)
PROGRAM=DEBUG
/SYSDEVS
SID=M91000
LOD=LP7A00
SWP=DM0800
```

```
//SOFTWARE
/PARAMETERS
SYSTEM=S7X.S001
SYMTAB=S7X.M001
DISP=32
POOL=2000
NTIM=60
MTIM=60
ITIM=384
ITLB=2C
TQFULL=1200
TQMIN=400
BATCHPRI=62
TERMPRI=60
PATCH=140
MODE=SIBP
/TABLES
JOBS=2
SHARE=10
TIMER=10
/FILES
SMD=DM0800,ENTRIES=2000
SYCSIZE=800
SGOSIZE=100
//END
```

7.7.2 SYSGEN Directive Example - 32/27

```
TITLE=32/27  SINGLE 300 MB DISC AND 32 MB DISCS  FLOPPIES  8-LINE ASYNC
//HARDWARE
/PARAMETERS
MACHINE=3227
/MEMORY
SIZE=64,TYPE=E,CLASS=S
SIZE=64,TYPE=H,CLASS=S
/CHANNELS
CONTROLLER=DM08,PRIORITY=06,CLASS=F,MPX=XIO,HANDLER=(H.EXIO,I)
DEVICE=(00),DTC=DM,HANDLER=(H.DP02,S),DISC=MH300
DEVICE=(02,2,2),DTC=DM,HANDLER=(H.DP02,S),DISC=CD032
CONTROLLER=CT7E,PRIORITY=13,CLASS=F,MUX=IOP,SUBCH=F
DEVICE=FC,DTC=CT,HANDLER=H.CTIOP,LINSIZ=80,PAGE=24
DEVICE=FE,DTC=CT,HANDLER=H.CTIOP,LINSIZ=80,PAGE=24
CONTROLLER=TY7E,PRIORITY=12,CLASS=F,MUX=IOP,SUBCH=C
DEVICE=(C0,8),DTC=TY,HANDLER=H.A8IOP,LINSIZ=80,PAGE=24
CONTROLLER=LF7E,PRIORITY=13,CLASS=F,MUX=IOP,SUBCH=F
DEVICE=F0,DTC=M9,HANDLER=H.FLIOP,DISC=FL001
DEVICE=F1,DTC=M9,HANDLER=H.FLIOP,DISC=FL001
DEVICE=F8,DTC=LP,SPOOL=(BL,RL),HANDLER=H.LPIOP
CONTROLLER=M910,PRIORITY=08,CLASS=F,MUX=XIO,HANDLER=(H.EXIO,I)
DEVICE=00,DTC=M9,HANDLER=(H.XMT,S)
CONTROLLER=NU00
DEVICE=00,DTC=NU,SPOOL=(BB,RB),SHR
/INTERRUPTS
PRIORITY=18,RTOM=(7F,06),PROGRAM=H.IPCL
PRIORITY=5F,RTOM=(7F,04),PROGRAM=H.IPIT,INTV
/TRAPS
PROGRAM=(H.IP00,H.IPAS,H.IP02,H.IP03,H.IP04,H.IP05,H.IP06)
PROGRAM=(H.IP07,H.IP08,H.IP09,H.IP0A,H.IP0C,H.IP0F,H.IP13)
PROGRAM=DEBUG
PROGRAM=H.IPHT
/SYSDEVS
SID=NU00
LOD=LP7EF8
SWP=DM0800
```

```
//SOFTWARE
/PARAMETERS
SYSTEM=S27.S001
SYMTAB=S27.M001
DISP=32
POOL=1800
NTIM=60
MTIM=60
ITIM=384
ITLB=2C
TQFULL=1200
TQMIN=400
BATCHPRI=62
TERMPRI=60
PATCH=800
/PARTITION
NAME=GLOBAL00,SIZE=64,STRTPG=C0,MAP=48
/TABLES
JOBS=2
SHARE=5
TIMER=4
/FILES
SMD=DM0800,ENTRIES=8000
SYCSIZE=600
SGOSIZE=100
//END
```

7.7.3 SYSGEN Directive Example - 32/87

```
TITLE=32/87 SINGLE 300 MB DISC  8-LINE ASYNC
//HARDWARE
/PARAMETERS
MACHINE=3287
/MEMORY
SIZE=64,TYPE=E,CLASS=S
SIZE=192,TYPE=S,CLASS=S
/CHANNELS
CONTROLLER=DM08,PRIORITY=06,CLASS=F,MPX=XIO,HANDLER=(H.EXIO,I)
DEVICE=00,DTC=DM,HANDLER=(H.DP02,S),DISC=MH300
CONTROLLER=M910,PRIORITY=08,CLASS=F,MUX=XIO,HANDLER=H.EXIO
DEVICE=00,DTC=M9,HANDLER=H.XMT
DEVICE=01,DTC=M9,HANDLER=H.XMT
CONTROLLER=TY7E,PRIORITY=1?,CLASS=F,MUX=IOP,HANDLER=H.IOPX,SUBCH=C
DEVICE=(C0,8),DTC=TY,HANDLER=H.A8IOP,LINSIZ=80,PAGE=24
CONTROLLER=CT7E,PRIORITY=1?,CLASS=F,MUX=IOP,SUBCH=F
DEVICE=FC,DTC=CT,HANDLER=H.CTIOP,IOQ=CONT,LINSIZ=80,PAGE=24
CONTROLLER=LF7E,PRIORITY=1?,CLASS=F,MUX=IOP,SUBCH=F
DEVICE=F0,DTC=FL,HANDLER=H.FLIOP,DISC=FL001
DEVICE=F1,DTC=FL,HANDLER=H.FLIOP,DISC=FL001
DEVICE=F8,DTC=LP,SPOOL=(BL,RL),HANDLER=H.LPIOP
CONTROLLER=NU00
DEVICE=00,DTC=NU,SPOOL=(BB,RB),SHR
/INTERRUPTS
PRIORITY=18,RTOM=(7F,06),PROGRAM=H.IPCL
PRIORITY=1C,RTOM=(7F,03)
PRIORITY=5F,RTOM=(7F,04),PROGRAM=H.IPIT,INTV
/TRAPS
PROGRAM=(H.IP00,H.IPAS,H.IP02,H.IP03,H.IP04,H.IP05,H.IP06)
PROGRAM=(H.IP07,H.IP08,H.IP09,H.IP0A,H.IP0C,H.IP0F,H.IP13)
PROGRAM=(H.IPHT,H.IP10)
PROGRAM=DEBUG
/SYSDEVS
SID=NU00
LOD=LP7EF8,IBP
SWP=DM0800
```

```
//SOFTWARE
/PARAMETERS
SYSTEM=S87.S001
SYMTAB=S87.M001
DISP=40
POOL=2400
NTIM=60
MTIM=60
ITIM=384
ITLB=1C
TQFULL=1200
TQMIN=400
BATCHPRI=62
TERMPRI=60
PATCH=800
/TABLES
JOBS=4
SHARE=4
TIMER=4
/FILES
SMD=DM0800,ENTRIES=8000
SYCSIZE=600
SGOSIZE=500
//END
```

7.7.4 Sample Resident User System Task

The example following is a task which could be configured in the resident operating system. The following restrictions apply to system-resident tasks:

1. They must be a single object module (programs cannot be linked). This effectively rules out FORTRAN programs.

2. They must not contain any CSECT, GLOBAL, or DATAPOOL directives. They may however freely use external Assembler references and definitions; i.e., they may reference subroutines in another system-resident task via the Assembler EXT/DEF mechanism.

3. They should never use exit; if they do, they cannot be reactivated, and will remain in memory regardless.

4. They do not have to be privileged, but they will not be able to alter any memory in the operating system unless they are made privileged. System-resident tasks are not loaded on page protected boundaries.

5. They cannot dynamically expand their address space.

6. They are free to use I/O task activation, inter-task communications, and other system services. Most memory management services are not usable.

7. They must not have been assembled using option 19 in the Assembler.

8. They must be compressed into the object file used as input to SYSGEN. (Note: SWAPR must be the last module in that compressed object file.)

```
* EXAMPLE OF USER RESIDENT TASK
*****************************************************************
*                                                               *
*                          U S E R T A S K                      *
*                                                               *
*          WHEN IN USE, THIS TASK WILL ALWAYS BE PRESENT        *
*          IN THE SYSTEM, AND CAN BE ACTIVATED BY THE           *
*          OPCOM !RESUME COMMAND.                               *
*****************************************************************
          PROGRAM  USERTASK
          LIST     NOMAC,NODATA
          M.EQUS                    MPX EQUATES MACRO
          M.MDE                     MACHINE DEPENDENT EQUATES MACRO
          M.TBLS                    SYSTEM TABLES
          M.SCE.                    STATE CHAIN INDEX EQUATES
          DEF      USERTASK         EXTERNAL DEFINE
          DEF      START
HC.CT     EQU      9B               HEAD CELL COUNT
QE.SF     EQU      0W               STRING FORWARD
QE.SB     EQU      1W               STRING BACK
BUFFERS   EQU      3
FILES     EQU      3
PRIORITY  EQU      03
          PAGE
```

```
*****************************************************************
*                                                               *
*                    H A N D  B U I L T  T S A                  *
*                                                               *
*****************************************************************
            BOUND     8W
USERTASK    EQU       $
TSA         EQU       $
            DATAW     1
            DATAW     SGINIT
            RES       6W
            GEN       1/1,7/0,24/START
            IFT       C.3227,32/77
PSDMAP      GEN       2/1,14/0,1/1,12/1,3/0
            GOTO      NOT77
32/77       ANOP
PSDMAP      GEN       2/1,14/0,1/1,13/1,2/0
NOT77       ANOP
            RES       320W
            DATAW     C'SYST'
            DATAW     C'EM  '
            DATAW     X'00000000'
            DATAW     0
            DATAW     TSA
            RES       38W
            DATAW     BBUFS,0,0,FATS,FPTS,0,0,0
            DATAW     BIAS,0,USEREND,START,0
            DATAB     0,0
            DATAB     BUFFERS
            DATAB     FILES
            DATAB     0,0,0,0
            DATAB     0,0,0,0,0,0,0,0
            DATAW     0
            DATAW     0
            RES       56W
*
*           FILE ASSIGNMENT TABLE AREA
*
FATS        EQU       $
            REPT      FILES
            DATAW     0,X'08000000',0,0,0,0
            ENDR
*
*           FILE POINTER TABLE
*
FPTS        EQU       $
            REPT      FILES
            DATAW     0,X'08000000',0
            ENDR
*
*           BLOCKING BUFFER AREA
*
BBUFS       EQU       $
            REPT      BUFFERS
            DATAW     X'08000000'
            RES       191W
            ENDR
```

```
*************************************************************************
*                                                                       *
*          W O R K I N G   S E C T I O N                                *
*                                                                       *
*************************************************************************
BIAS        EQU     $
START       EQU     $
            LA      R1,TYPEBLK
            SVC     1,X'3F'
            SVC     1,X'54'
            ZR      R5
            ZR      R6
            ZR      R7
            BU      START
MESSAGE     DATA    C'"M"JTHIS IS A SYSTEM-RESIDENT TASK'
TYPEBLK     GEN     12/32,20/B(MESSAGE)
            DATAW   0,0
            LPOOL
USEREND     EQU     $
            PAGE
```

```
*****************************************************************
*                                                               *
*       THIS ROUTINE IS EXECUTED AT SYSTEM TIME ONLY.           *
*       IT IS OVERLAYED BY THE NEXT MODULE.                     *
*                                                               *
*****************************************************************
SGINIT  EQU     $
        M.EIR                           INIT EP MACRO
        LA      R1,C.FREE               GET HEAD CELL ADDRESS
        LW      R2,C.FREE               GET ADDRESS OF ENTRY TO UNLINK
        LI      R3,-1
        ARMB    R3,HC.CT,R1             DECREMENT HEAD CELL COUNT
        LW      R1,QE.SF,R2             DO UNLINK
        LW      R3,QE.SB,R2
        STW     R1,QE.SF,R3
        STW     R3,QE.SB,R1
        LA      R1,C.SUSP               ADDR OF NEW HEAD CELL
        ABM     7,HC.CT,R1              INCREMENT HEAD CELL ENTRY COUNT
        LW      R3,QE.SB,R1             LINK FREE DQE INTO SUSP STATE
                                        CHAIN
        STW     R2,QE.SB,R1
        STW     R2,QE.SF,R3
        STW     R3,QE.SB,R2
        STW     R1,QE.SF,R2
        LA      R1,TSA                  INITIALIZE CPIX
        STW     R2,T.PRNO,X1            SET UP OUR DQE ADDR IN TSA
        LA      R3,DQEPROTO             GET ADDRESS OF PROROTYPE DQU
        LA      R6,T.MIDL,X1            GET MIDL FOR DQE
        STW     R6,DQE.MSD,X3           FOR 7X PUT IN DQE
        LB      R6,DQE.NUM,X2           GET TASK NUMBER
        TRR     R6,R7
        SLL     R7,2
        ORMW    R7,PSDMAP
        STW     R7,PSDMAP
        LI      4,DQE.SIZE-2W           MOVE PROROTYPE INTO DQE
        TRN     R4,R4
        TRR     R2,R1
        ADI     R3,2W
        ADI     R2,2W
LOOP    LW      R7,0,R3
        STW     R7,0,R2
        ADI     R3,1W
        ADI     R2,1W
        BIW     R4,LOOP
        ABM     31,C.ACTSEQ             BUILD TASK NUMBER
        SLL     R6,24
        ORMW    R6,C.ACTSEQ
        STW     R6,DQE.NUM,R1
        M.XIR   TSA
```

```
*****************************************************************
*                                                               *
*        D I S P A T C H  Q U E U E  E N T R Y                  *
*                                                               *
*****************************************************************
DQEPROTO DATAW    0,0
         DATAB    PRIORITY,PRIORITY,PRIORITY
         DATAB    X'13'
         DATAW    0
         DATAW    C'SYST'
         DATAW    C'EM'
         DATAW    C'USER'
         DATAW    C'TASK'
         DATAW    0
         DATAW    0
         DATAW    0,X'50000200'
         DATAW    0,0,0,0
         DATAW    0
         REZ      7W
         DATAW    $,$-1W,0
         DATAW    $,$-1W,X'8000'
         DATAW    $,$-1W,0
         DATAW    $,$-1W,0
         DATAW    0
         GEN      8/0,24/TSA
         DATAW    0,0,0,0
         REZ      6W
         END
```

8. THE MPX-32 SYSTEM DEBUGGER (H.DBUG)

The MPX-32 System Debugger is provided on the master SDT. It can be used to debug the resident operating system as well as SYSGEN'd user interrupt and I/O handlers. The Debugger is not SYSGEN'd as part of the resident MPX-32 starter system, but it can be included in any user configuration of the resident system by using the SYSGEN directive PROGRAM = DEBUG. The System Debugger adds approximately 4KW to the size of the resident system.

The System Debugger can only operate in privileged mode because it is using stand alone I/O. Therefore, all 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, CALM, etc., and traced by the Debugger.

Unusual instruction sequences (branch increment with positive register contents, load file/store.file on a non-file boundary, etc.) are flagged with a question mark following the instruction text. These are attempts to make a user aware of a possible problem.

The Debugger always starts in symbolic mode, which allows addressing by base name plus offset. The AB (Absolute) command can be used to switch to absolute addressing. The SY (Symbolic) command switches back to symbolic addressing. (Absolute or symbolic modes apply to all address displays.)

8.1 Using the Debugger

8.1.1 Arithmetic and Special Operators

The Debugger uses several characters as unique operators allowable in any command or expression:

Character	Usage
Colon (:)	Last displayed contents (current value)
Asterisk (*)	Indirect address if first character in a field
Slash (/)	Current total
G thru Z	Symbolic base values
(R)	Register designator
Semi-colon (;)	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 left subtracted from field on right
Asterisk (*)	Field on left multiplied by field on right
Slash (/)	Field on left divided by field on right
Ampersand (&)	Field on left is OR'd with field on right
At-sign (@)	Field on left is AND'd with field on right
Left Angle Bracket (>)	Field on left is shifted right by the count in the right side field
Right Angle Bracket (<)	Field on left is shifted left by the count in the right side field
Apostrophe (')	ASCII text delimiter
Backslash (\)	Subfield delimiter

8.1.2 Special Functions

The Debugger normally expects to read a two-character command and options; however, some special functions may be input in lieu of the normal commands:

Function	Result
^	Display previous location.
*	Display location indirect to current location.
carriage return	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.

8.1.3 Execution Breakpoints

Eight 'fixed' breakpoints can be defined. In addition, a one-shot breakpoint can be established for the directed execution commands (GO, CO, or CT).

8.1.4 Debugger Bases

The Debugger uses a range of one-character bases to provide symbolic references to memory. Once established, the base characters (characters G thru Z) may be used in any command or arithmetic expression. The following bases are initialized by the Debugger when loaded with MPX-32:

Base	Module
G	H.LODR
H	H.SOUT
I	H.EXEC
J	Available
K	Available
L	H.ALOC
M	H.FISE
N	TSA start
O	Available
P	Available
Q	IP06 (SVC)
R	J.SWAPR
S	X'78000' (User DEBUG)
T	H.TSM
U	Available
V	Available
W	H.MONS
X	C.xxxx Region
Y	H.IOCS
Z	Available

Note that 40I is equivalent to 40+I or I+40.

8.1.5 Base Characters

The characters 'G' thru 'Z' can be set to any value by the BA (Base) command, i.e.,

```
BA G 427FO
```

would cause G to be henceforth known as location 427FO, and G would appear in all address displays in place of 427FO. The user may use G in any command and it will be processed as though he had typed in 427FO.

8.1.6 Operator Restrictions

Arithmetic expressions are evaluated left to right.

Operators may appear in any command and be as long as expressions may require to state. For example:

```
AR D9C35FFF,@3S00000,>14
```

This would result in a hex 18.

```
AR *34595,<2, +Z, +3000
```

If location 34595 contained 500 (hex) and Z was set to 30,000, the above expression would have a value of 34,400.

```
DM 0 + 34C2 -1>2
```

The location specified by 0 was previously set to 30,000, so the DM location would be CD30 (30,000 + 34C2 -1 and right shifted 2 bits).

The above examples illustrate expressions that may be used in any appropriate command.

8.1.7 Expressions

The Debugger will process any expression given in a left to right manner, i.e.,

```
AR 2*32+G/2
```

would break down to two times 32 plus the value of G. That total is divided by 2 and the answer is typed.

8.1.8 Registers

The contents of a register can be used in any expression by enclosing the register number in parentheses, i.e.,

```
DM (1)
```

would display the contents of R1.

The DR command can be used to display the registers and the CR command can be used to modify the registers.

8.1.9 Indirection

Indirection must precede a field, i.e.,

```
DM *0
```

would cause the value at location 0 to be obtained and used as an address to display.

8.2 Accessing the Debugger

If the user configures the System Debugger as part of a resident system, the Debugger is automatically accessed by BOOTxx when the configured system is installed from the user SDT. (The installation process is described in Chapter 2.) The Debugger is activated by BOOTxx. The Debugger input prompt is a double angle bracket:

```
>>
```

At this point, the user can issue Debugger commands from the OPCOM console and make patches in the memory resident image of the system. BOOTxx copies the patched version to disc as the disc image. The Debugger is terminated by a TE (Terminate) command. It returns control to BOOTxx, which continues building the system as described in Chapter 2.

There are two other ways the Debugger can be accessed. A privileged user task can access it by coding a branch and link through the communications region variable that points to the Debugger (*C.DEBUG):

```
BL  *C.DEBUG
```

Branch and link is the technique used by OPCOM.

Or a Debug command is available in OPCOM:

```
TSM> OPCOM DEBUG
```

The System Debugger uses stand alone drivers to perform I/O. The Debugger routes listings to the printer configured as LP7A and it gets commands from the terminal or teletype configured as TY7E (normally the OPCOM console). This means that although any terminal user can issue the OPCOM DEBUG command, once the System Debugger gains control, its prompt is displayed on the OPCOM console and it accepts commands only from that device.

8.3 Debugger Commands

When the Debugger is entered, it initializes bases and sets symbolic mode. It is then ready to receive and process commands.

The command syntax requires the first two characters be input unless special operators are used. Certain commands will contain one or more fields of additional information, separated by a comma, i.e.:

Has value of up to eight characters. Fewer than eight result in right adjusted values.

Has value, arithmetic operator, and a base character.

Special operators representing current PC, current value, current total, etc.

Special indirect operator for any operator.

Debugger commands are summarized below and described in detail in pages which follow.

Command	Function
AB	Displays all addresses as numeric.
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 via a Branch and Link).
CM	Changes the contents of memory to a new value.
CO	Continues tracing or execution from a breakpoint which was set with a BR command.
CR	Changes the value of a user register to a new value.
CT	Continues processing, setting a one-shot breakpoint at a specified address which terminates the trace function.
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.
DR	Displays all eight registers.
DS	Displays memory locations in instruction format within a specified address.

DT	Dumps the Event Trace Table to the printer. Note: This command 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.
LB	Lists all breakpoints.
LP	Sets line printer output mode.
LT	Displays a list, on the console, of the current mobile event trace points.
PS	Displays last known user PSW and condition codes.
RE	Remaps the Debugger to the map associated with a specified 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 command, to a specified value.
SM	Sets the mask (left-justified hexadecimal number) for the SE command. If not specified, defaults to X'FFFFFFFF'.
SP	Dumps CPU scratchpad ram locations.
SY	Displays addresses as displacements from bases (see BA command).
TE	Same as BY command.
TR	Traces user programs and displays each instruction after execution.
TS	Terminates trace initiated by the TR command.
TY	Sends output to a terminal and resets Echo mode.

8.3.1 AB (Absolute) Command

The AB command is used to display all addresses as numeric.

Syntax:

```
AB
```

8.3.2 AD (Address) Command

The AD command is used to display the low and high limits of a task's address space.

Syntax:

```
AD
```

Example:

```
>>AD
START = xxxxxxxx END = yyyyyyyy
>>
```

8.3.3 AR (Arithmetic) Command

The AR command is used to evaluate an arithmetic expression and display its value.

Syntax:

```
AR expr
```

expr is the arithmetic expression to be evaluated and can be as long as required to state.

Response:

The Debugger processes the expression in its left to right order and the answer is displayed.

8.3.4 AS (Assemble Instruction) Command

The AS command converts an assembly language instruction to its hexadecimal equivalent.

Instruction groups supported are memory-reference and register-register.

I/O instruction group is not supported at this time.

Also, the Assembler mnemonics used in the syntax for this command are abbreviated to only four characters rather than five characters. Use the DI command to obtain a display of the valid 4-character mnemonics available for use with the AS command.

Syntax:

AS opcode [,reg] [,offset] [,index]

opcode is the 4-character Assembler mnemonic

reg is a number in the range 0-7 or the characters R0-R7

offset is a hexadecimal number or an expression

index is the number 1, 2 or 3

Response:

The 8-digit hexadecimal number corresponding to the instruction is displayed.

8.3.5 . BA (Base) Command

The BA command is used to define a user base (add its name to the internal base definition table), delete a user base name from the base table, or redefine a user base (change the value specified in the base name's definition).

Up to twenty bases are allowed. See Section 8.1.4.

Syntax:

BA base [addr]

base is a 1 character alphanumeric base name (see Section 8.1.4).

addr is the logical address for the base. If not supplied, the specified base name is deleted. If "addr" is supplied and "base" is already defined, "base" is redefined to represent "addr".

8.3.6 BR (Breakpoint) Command

The BR command is used to set a breakpoint at a specified address. 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

addr is the address at which the breakpoint is set.

Response:

A breakpoint is set at the specified address and remains until cleared with the DE command. Upon execution of the breakpoint, BRK@ADDR is output along with the contents of the registers to indicate which breakpoint was executed.

8.3.7 BY (Bye) Command

The BY command is used to exit the Debugger when entry was via a branch and link. See also the TE command.

Syntax:

BY

Response:

The Debugger returns control to OPCOM or the calling program.

8.3.8 CM (Change Memory) Command

The CM command is used to change the contents of one or more words beginning at a specified address to a new 32-bit value.

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.

Response:

The specified address is changed to either a right adjusted hexadecimal value or a left adjusted blank-filled ASCII text word.

8.3.9 CO (Continue) Command

The CO command is used to continue tracing or execution from a breakpoint. If the user is in trace mode (entered via a TR command), tracing continues. If the user is not in trace mode, execution continues (see GO command).

Syntax:

CO [addr[,stop]]

addr is the address location from which program execution with optional trace will continue. If not specified, the program resumes from the last known location.

stop is the address location at which execution and tracing will terminate.

Response:

The Debugger resumes execution and will not exit while TR is active.

8.3.10 CR (Change Register) Command

The CR command is used to modify 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 can be used to skip the user register corresponding to the missing value, leaving its contents unchanged. If user register R7 has been altered or skipped and an unused value remains, it is ignored.

Response:

The contents of the specified registers are displayed.

8.3.11 CT (Continue then Terminate) Command

The CT command is used to continue 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 at which an optional one-shot breakpoint will be set. Tracing is terminated and the users program reentered. If not specified, return to the users program will be at the last known program counter value.

Response:

The user program is reentered.

8.3.12 DE (Delete) Command

The DE command is used to delete a breakpoint which was set with a BR command and restore user instructions to their original locations.

Syntax:

DE addr

addr is the address at which the breakpoint is deleted and the users instruction is restored to its original location.

8.3.13 DI (Display Instruction) Command

The DI command is used to display any memory locations within a specified address range, in instruction format, one line at a time (same as DS command).

Syntax:

DI add1 [,add2]

add1 is the address at which the display starts.

add2 is the address at which 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.

Response:

Memory locations are displayed in instruction format within the specified address range until the display is terminated.

8.3.14 DM (Display Memory) Command

The DM command is used to display all memory locations within the specified address range.

Syntax:

DM add1[,add2] [,number]

add1 is the address where the display will start.

add2 is the address where the display will end. If not specified, the ending address defaults to the starting address, i.e., add1 is the only location displayed.

number is the number of words to be displayed per line on the console (minimum 1, maximum 8). Default: 4

Response:

The specified memory locations are displayed. If DM is used with no parameters, memory locations are displayed 24 lines at a time. A carriage return will continue the display, any other character entered will terminate the display.

8.3.15 DR (Display Register) Command

The DR command is used to display the memory locations of registers (R0-R7).

Syntax:

DR[reg]

reg is a user register (R0-R7). If not specified, all registers are displayed.

Response:

Memory locations of the requested registers are displayed.

8.3.16 DS (Display Symbolic) Command

The DS command is used to display any memory locations within a specified address range, in instruction format, one line at a time (same as DI command).

Syntax:

DS add1[,add2]

add1 is the address at which the display starts.

add2 is the address at which 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.

Response:

Memory locations are displayed in instruction format within the specified address range until the display is terminated.

8.3.17 DT (Display Event Trace) Command

The DT command is used to dump the Event Trace Table to the printer.

Caution: This command assumes memory partition at hex 78000 and requires reassembly of the entire resident source with Event Trace enabled.

Syntax:

DT

Response:

The Event Table is output to the line printer and Event Trace is disabled until printing is completed.

8.3.18 DU (Dump) Command

The DU command is used to output 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]]

If no parameters are specified, the entire O.S. area is output.

start is the memory address at which the dump will start.

stop is the memory address at which the dump will stop. If not specified, the end of the O.S. is assumed.

8.3.19 EC (Echo) Command

The EC command is used to generate hard copy output when using a CRT as the terminal device.

Syntax:

EC

Response:

Output is listed, line by line, on the line printer for every carriage return initiated on the terminal.

To terminate the EC command, reset the TY command.

8.3.20 ET (Enter Event Trace Point) Command

The ET command is used to place an event trace point within either resident or non-resident code without requiring reassembly of a program.

Prerequisites for use 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 command) cannot be set at the same address location.

Syntax:

ET addr

addr is an absolute or relative address with a symbolic base

Response:

An event trace point is placed at the specified address.

8.3.21 GO (Go) Command

The GO command is used to transfer control to the user task, optionally setting a one-shot trap at a specified address.

Syntax:

GO[start [,stop]]

If no parameters are specified, the program is entered at the last known user PSW.

start is the address at which the program is to start executing.

stop is the address at which the program stops. 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.

8.3.22 LB (List Breakpoint) Command

The LB command is used to list all breakpoints.

Syntax:

LB

Response:

All active fixed breakpoints (breakpoints which were set via a BR command) are displayed.

8.3.23 LP (Line Printer) Command

The LP command is used to direct output to the line printer.

Syntax:

LP

Response:

All further I/O is directed to the line printer for hard copy output. If the user is tracing, no request is made for input after each instruction. Therefore, a breakpoint or optional STOP address must be set to terminate the trace.

8.3.24 LT (List Mobile Event Trace Point) Command

The LT command is used to list current mobile event trace points within resident or non-resident code.

Prerequisites for use are:

1. The system must have been SYSGENed with a static partition for the Event Trace Table.

2. Both H.DBUG (Symbolic Debugger) and H.IP06 (SVC Trap Handler) must have been assembled with TRACE set true.

Syntax:

```
LT
```

Response:

A list is displayed on the console of all current mobile event trace points.

8.3.25 PS (Program Status) Command

The PS command is used to display the user Program Status Word (PSW) and condition codes (cc's).

Syntax:

```
PS
```

Example:

```
>>PS
PSW@  A002F830   CC=0100

A102F830  0001B10C  00000000  0000002A          *...0.....**
30312F30  372F3830  44454255  47202020          *01/07/80DEBUG *
>>
```

8.3.26 RE (Remap) Command

The RE command is used to remap the Debugger to the map associated with a program.

Syntax:

RE[number]

number is the Dispatch Queue number in the range 0-FF to which remapping will be associated. If not specified, remapping defaults to the current program.

8.3.27 RT (Remove Event Trace Point) Command

The RT command is used to remove a mobile event trace point within resident or non-resident code.

Prerequisites for use are:

1. The system must have been SYSGENed with a static partition for the Event Trace Table.
2. Both H.DBUG (Symbolic 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

Response:

The mobile event trace point at the specified address is removed.

8.3.28 SE (Search Equivalent) Command

The SE command is used to compare specified words to a specified value. Each word is ANDed with the Search Mask (see SM command) before being compared to the value. Each bit in the range is listed.

Syntax:

SE value, start, stop

value is the value to which each word will be compared.

start is the address from which the search will begin.

stop is the address at which the search will end.

Example:

```
>> SE
 0      00000000                          *....*
>>
```

8.3.29 SM (Set Mask) Command

The SM command is used to set the mask for the SE command. 'Mask' 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, defaults to the previously entered value, or if none, X'FFFFFFFF'.

8.3.30 SP (Scratchpad Dump) Command

The SP command is used to output to the terminal the contents of the scratchpad locations.

Syntax:

SP [loc]

loc — is the CPU scratchpad address 0-FF. If not specified, all scratchpad locations are displayed.

Example:

```
>> SP
300  00000000  00000000  00000000  00000000  *................*
310  00000000  00000000  00000000  00000000  *................*
320  0E690800  0E690801  0E690802  0E690803  *.i...i...i...i..*
etc.
>>
```

Twenty-three lines of text are output at a time. A carriage return continues output, entering any other character terminates output.

8.3.31 SY (Symbolic) Command

The SY command is used to display all addresses as displacements from bases (see BA command).

Syntax:

SY

8.3.32 TE (Terminate) Command

The TE command is used to exit the Debugger when entry was via a branch and link. If entry was via a breakpoint, the CO command is simulated.

See also the BY command.

Syntax:

TE

Response:

The Debugger returns control to OPCOM or the calling program.

Note: The TE command should not be used to exit from a breakpoint (see the CO and CT commands).

8.3.33 TR (Trace) Command

The TR command is used to execute and display 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 command. 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, and an equal sign (=) causes the hexadecimal equivalent of the instruction just executed to be displayed. Any other character will cause the Debugger to prompt for a command.

Symbols established by the BA command are used for display purposes if the SY command was set.

Syntax:

TR [start[,stop]]

start is the address of the first user instruction to be traced. Defaults to $ (current PSD value).

stop is the address of the last user instruction to be traced. If not specified, tracing continues without bounds.

Example:

```
>> TR
N+7830          LF   R0,N+1820     R0=D102A7DC
                                   R1=FFFFFFF0
                                   R2=0
                                   R3=2A
                                   R4=30312F30
                                   R5=372F3830
                                   R6=44454255
                                   R7=47202020
N+7834          ZR   R7            R7=0
>>
```

8.3.34 TS (Trace Stop) Command

The TS command is used to exit the trace mode initiated by the TR command.

Syntax:

TS

Response:

All further I/O will be directed to the user's terminal.

8.3.35 TY (Terminal) Command

The TY command is used to direct output to a terminal and to reset EC (Echo) mode.

Syntax:

TY

9. ON-LINE SYSTEM PATCH FACILITY (J.INIT)

J.INIT provides for temporary or permanent patching of the MPX-32 resident image.

9.1 General

The System Initialization Task (J.INIT) performs three basic functions:

- o Initializes disc allocation maps and spooled I/O directories (start-up only).
- o Activates J.TSM and other start up programs.
- o Processes patch directives from the file M.PATCH.

Disc allocation maps for all on-line discs are rebuilt during start-up from data contained in the SMD, M.SID and M.SOD files. If any of these definitions are in conflict, an allocation overlap message specifying the starting sector of the overlap is written to the console. In the event of file overlaps, the System Administrator should delete M.SID and M.SOD before rebooting. If the messages persist, the FILEMGR should be invoked and the LOGS (sequential) function should be used to determine which files conflict. The system should not be used until the overlaps are eliminated.

After processing the patches and the disc allocation, J.INIT will activate J.TSM and any other start-up program specified by SYSGEN (effective with MPX-32 Release 1.2).

The patch file (M.PATCH) should contain valid patch directives as described in Section 9.2. The patch file is maintained by the text editor and should be a STORED (blocked, uncompressed) file. An associated symbol table file is built by SYSGEN in conjunction with the SYMTAB directive. To generate a listing of the resident modules and unique names given to duplicate handlers, specify the option SYM with the /O directive in the M.PATCH file. In addition, the SYSGEN PATCH directive should be used to define the size of the patch area.

Patch processing terminates when the EXIT (/E) directive is encountered in the patch input file. On termination, an audit trail of all patches specified will be generated on the SLO file. The audit trail listing can be suppressed by specifying the option NPR with the OPTION (/O) directive.

The patch program accepts directives to control processing. The general format of a directive is:

$$/d\ f_2\ f_3\ f_4\ f_n$$

where:

/d	is the directive name which must be followed by one or more spaces. This is called field 1 for error messages.
f_2 - f_n	are fields which contain the names, values and special symbols processed by the directive. Each field must be separated by one or more spaces or a comma.

Although the structure of J.INIT allows most fields to be specified on statement, only those of general use are described.

9.1.1 Dedicated Names

Dedicated names used by J.INIT are:

$	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	1-8 ASCII characters, one of which must be non-numeric
value	1-8 hexadecimal digits; leading zeros need not be specified. Only whole words are generated.
address	1-5 hexadecimal digits; leading zeros need not be specified. Must be word resolution (bits 30-31=0).

9.2 Patch Directives

The following directives are available:

/B	Define a base address
/D	Define a named value
/C	Change the contents of a memory location
/G	Go to the patch area from a specified memory location
/$	Enter a value into the patch area
/R	Return from the patch area
/F, /N, /T	Process patch directives conditionally
/P	Define a patch area
/;	Comment only
/E	Exit
/O	Select patching options

9.2.1 Define a Base Address (/B)

The /B directive allows a name to be equated to a base address. This definition is inserted in the internal symbol table and may be referenced by subsequent directives.

Syntax:

/B name address

where:

name is equated to "address"

(or)

/B $name_1$ $address_2$

where:

$name_1$ is equated to the address value of "$name_2$"

9.2.2 Define a Named Value (/D)

The /D directive allows a name to be equated 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 may be referenced by subsequent directives.

Syntax:

/D name value

where:

name is equated to the address of "value" in the patch area

(or)

/D $name_1$ $name_2$

where:

$name_1$ is equated to the address value of "$name_2$"

9.2.3 Change the Contents of a Memory Location (/C)

The /C directive allows any location in memory to be changed to a specified value.

Syntax:

/C address value

where:

the specified "value" is inserted at the specified "address"

(or)

/C address value R

where:

the specified "value" is inserted, relative to the specified "address", at the specified "address" (the value of "address" is added to the address field of "value")

(or)

/C address value -R

where:

the specified "value" is inserted, relative to the specified "address", at the specified "address" (the address field of "value" is subtracted from "address" to form the address field of "value")

(or)

/C name address value

where:

the specified "value" is inserted at the specified "address", relative to "name" (the value of "name" is added to "address" to form the actual address)

(or)

/C name address value R

where:

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)

(or)

/C name address value -R

where:

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")

(or)

/C $name_1$ address value $name_2$

where:

the value of "$name_2$" is added to "value" and inserted in "address" relative to "$name_1$" (the value of "$name_1$" is added to "address" to form the actual address

9.2.4 Go to the Patch Area from a Specified Memory Location (/G)

The /G directive allows an unconditional branch to the patch area to be inserted 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

where:

an unconditional branch is inserted to the patch area at the specified "address"

(or)

/G name address

where:

an unconditional branch is inserted to the patch area at "address", relative to "name" (the value of "name" is added to "address" to determine the actual address)

(or)

/G name address value

where:

an unconditional branch is inserted to the patch area at "address", relative to "name" (the branch is to the next free patch location plus some other number of bytes (word resolution)). The offset area is thus reserved. The basic form, no "name" field, can also be used.

9.2.5 Enter a Value into the Patch Area (/$)

The /$ directive is similar to the /C directive except that the value is inserted into the next free location of the patch area.

Syntax:

/$ value

where:

value is inserted into the next free patch location

(or)

/$ value R

where:

value is inserted, relative to the address of the next free patch location, into the next free patch location (the value of $ is added to "value")

(or)

/$ value -R

where:

the address field of "value" is subtracted from $ to form the new address field of "value" and stores this in the next free patch location

(or)

/$ name value

where:

"value" is inserted, relative to "name" (the value of "name" is added to "value") into the next free patch location

9.2.6 Return from the Patch Area (/R)

The /R directive allows an unconditional branch back to the instruction (plus 1 word) produced by the last /G directive encountered. An offset may, optionally, be specified to reserve a number of patch locations immediately following the branch back.

Syntax:

/R

where:

an unconditional branch is inserted to the location (plus 1 word) containing the last branch generated by a /G directive

(or)

/R value

where:

an unconditional branch is inserted to the location (plus 1 word) containing the last branch generated by a /G directive ("value" bytes of the patch area (word resolution) are reserved in the patch area. This reserved area follows the generated branch.)

9.2.7 Conditional Directive (/F, /N, /T)

The /F, /N, and /T directives are used to allow skipping the processing of other directives based upon the presence or absence of a specified name in the symbol table. This allows a general patch deck to be created which will attempt to modify only those modules included in the MPX-32 resident image.

Syntax:

/F $name_1$ $name_2$

where:

if "$name_1$" is false (not defined in the symbol table) discontinue directive processing until a /N directive containing the name "$name_2$" is encountered

(or)

/T $name_1$ $name_2$

where:

if "$name_1$" is true (defined in the symbol table) discontinue directive processing until a /N directive containing the name "$name_2$" is encountered

(or)

/N name

where:

the /F or /T skip sequences are terminated

9.2.8 Define a Patch Area (/P)

The /P directive is used to define a temporary patch area or to append patches to the patch area defined by SYSGEN. It is intended to be used during debugging and should never be used when saving (see /O directive) patches.

Syntax:

/P address value

where:

a patch area is defined starting at location "address", "value" (word resolution) bytes long. Since no attempt is made to protect this area, it should be some area of the resident image not used during debug operations.

(or)

/P CUR

where:

subsequent patches are added to those entered during a previous patch run

9.2.9 Comments (/;)

The /; directive may be included on any patch directive as a delimiter. The total directive may be designated as a comment by the use of this directive.

Syntax:

/; text

9.2.10 Exit (/E)

The /E directive is used to terminate directive processing.

Syntax:

/E

9.2.11 Select Patch Options (/O)

The /O directive is used to specify punch options for controlling the processing of subsequent directives.

Syntax:

/O $name_1$ $name_2$ $name_n$

where:

$name_1$, $name_2$ and $name_n$ are the option names. A single /O directive may contain more than one "name" and any number of /O directives may be used.

The available options are:

"name"	Use
NAM	Informs J.INIT that the definitions in file M.PATCHN should be merged into the internal symbol table. This option should be specified only once in a single run.
NHE	Informs J.INIT not to halt if patch errors are detected. If not specified, any patch error causes a halt; depress START switch to continue.
NPR	Informs J.INIT that a patch listing is not to be produced.
SAV	Informs J.INIT that all patches are to be saved in the M.PATCH file. Any error will cause the M.PATCH file to be cleared.
SYM	Directs J.INIT to produce a listing of an internal symbol table. This should not be present on the first directive of a patch deck.

9.3 Entry Conditions

Calling Sequence:

J.INIT is activated by J.SWAPR at start up.

J.INIT may also be activated from TSM or OPCOM once the system is running. If activated by this method, only the PATCH function is performed, not the disc initialization.

The patch processing can be inhibited by setting control switch 2 via the front panel. In addition, terminal initialization can be inhibited by setting control switch 3.

9.4 Exit Conditions

Return Sequence:

```
M.CALL    H.MONS,18    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:

(1) 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.

(2) J.INIT may output error messages along with the audit trail listing. All error messages are preceded and followed by seven asterisks (*******). Possible error messages follow:

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 "name" in the internal symbol table and it is currently in the table.

END OF FILE ON M.PATCH

An attempt has been made to save the source image of a patch on file M.PATCH and it is full (limit = 900 images).

UNABLE TO ALLOCATE M.PATCH

If file M.PATCH does not exist, J.PATCH attempts to create it. This message indicates that sufficient disc space was unavailable (100 blocks).

UNABLE TO ALLOCATE INPUT DEVICE

MPX-32 denied the request to allocate the patch input device.

SAVE AND ADD PATCHES SPECIFIED

Indicates that a /O SAVE option was followed by a /P CUR directive.

PATCH ERRORS DETECTED

Output at the end of the audit trail if any patch errors were detected. Also output to the console teletypewriter.

(3) J.INIT may output the following error messages to the console:

PATCH PROCESSING ERRORS - LIST PATCHES

Output if any patch errors were detected.

ERROR IN DYNAMIC SLO ALLOCATION - AUDIT TRAIL INHIBITED

The SLO file could not be allocated - the NPR option is forced and processing continues.

J.INIT - UNRECOVERABLE I/O ERROR ON DEV xxxx DURING RECREATION OF ALLOCATION MAP - DEVICE NOT ALLOCATED

An I/O error occurred when trying to allocate device xxxx - allocation of that device is bypassed.

BOTH M.SID/M.SOD DELETED/RECREATED

Invalid UDT index used on reallocation of SLO/SBO file or disc overlap on reallocation.

J.INIT - ZERO DETECTED AS UDTX ON ALLOCATION - MAP DISCS NOT RE-INITIALIZED

A zero UDT index was found in the allocation table - initialization continues.

SMD MAP ERRORS --- SYSTEM DISC MAP NOT RE-INITIALIZED

An unrecoverable I/O error occurred when trying to read the allocation map.

J.TSM ACTIVE --- DISC MAPS NOT RE-INITIALIZED

J.TSM was found to be active - disc allocation maps could not be recreated.

PERMANENT FILE ALLOCATION OVERLAP --- SORTING SECTOR ADDRESS nnnnn

An overlap occurred when reallocating disc space.

9.6 Examples

(1) 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
```

Note 1: Changes location 2154 of H.IOCS to a LI R5,X'3331'.

Note 2: Changes location 574 of H.IOCS to a BU to location 1003 of H.IOCS.

(2) Inserting into the patch area

```
/O NPR NHE NAM SAV              See Note 1
/G H.MONS 100                   See Note 2
/$ CB050001                     See Note 3
/$ H.MONS F20005E5              See Note 4
/$ EE000009 R
/R
/$ 0000 0000
/O SYM                          List symbol table
/E                              End of patches
```

Note 1: No print, no halt on error, get module definitions, save patches.

Note 2: Branch to the patch area from location 100 of H.MONS.

Note 3: CI R6, 1 inserted in the next free location of patch area.

Note 4: BCF EQ,5E5 of H.MONS inserted.

10. MEMORY-ONLY MPX-32

10.1 Overview

Memory-only MPX-32 (MPX-32/M) is a condensed version of the MPX-32 operating system that does not support any file structure capability, i.e., MPX-32 utilities (MPX-32 Reference Manual, Volume 2) are not compatible with MPX-32/M systems. MPX-32/M is designed for use in real-time environments where all tasks reside permanently in memory and limited operator communications is required. System generation is accomplished under the control of a fully operational MPX-32 host environment with memory-only remote loading on the targen machine conducted through a magnetic tape or floppy disc mechanism.

System installation and initial task loading/activation is accomplished by means of a system distribution medium of magnetic tape of floppy disc (SDT) in a manner similar to MPX-32. The user has the option of loading a modified version of the operator communications (OPCOM) task from the SDT. This allows limited communication with the memory-only system via the operator's console. For further details of OPCOM usage on a MPX-32/M system, see Section 10.6.

Other devices may be configured, provided they are properly allocated to the appropriate tasks and are not accessed through a file structure mechanism. Dynamic task activation from tape or floppy disc is also permitted. Tasks may share memory by means of static memory partitions or shared CSECT regions only. Dynamic memory partitions are not supported under MPX-32/M.

If operating system size is not a constraint, a memory-only environment can be created with a fill MPX-32 system. After activating the appropriate tasks in a normal manner, all discs are turned off or placed offline. This, in effect, causes all file access operations to result in an unrecoverable I/O error when an attempt is made to read the System Master Directory (SMD). These errors can then be handled as the user sees fit.

10.2 Memory Allocation

The following are memory allocation requirements for MPX-32/M systems with or without the System Debugger under an 8KW (32/7x) and 2KW (32/27) map granularity.

	Memory Requirements	
Module	8KW Map	2KW Map
MPX-32/M	24KW	22KW
MPX-32/M W/DEBUG	24KW	26KW
SYSBUILD	8KW	2KW
OPCMM	8KW	6KW

10.2.1 Shared Memory

MPX-32/M supports the concept of shared memory in the form of GLOBAL common, DATAPOOL and shared CSECT regions. Dynamic shared memory is not supported. Tasks are compiled and cataloged in a full MPX-32 environment just as they are for a disc-based system.

In the case of GLOBAL common and DATAPOOL, a static partition must be defined at SYSGEN by means of a PARTITION directive. Since such tasks will be operating in a system environment different from the one in which they were cataloged, the static partition in the memory-only environment must have the same partition name, protection granule size and starting logical protection granule definitions as that for the host system. The physical starting map number need not be identical for the two systems. For example on a SYSTEMS 32/27, a DATAPOOL task that was cataloged under a 128KW system with the following static partition definition

```
/PARTITION
NAME=GLOBAL55,SIZE=8,STRTPG=F8,MAP=62
```

could contain the following partition definition for a 64KW MPX-32/M operating environment

```
/PARTITION
NAME=GLOBAL55,SIZE=8,STRTPG=F8,MAP=30
```

Additionally, all tasks wishing to use GLOBAL common or DATAPOOL in the MPX-32/M environment must call the INCLUDE system service (M.INCL or equivalent subroutine call) on their own behalf prior to accessing the shared memory. This is the mechanism through which the memory partition is mapped into the task's address space under MPX-32/M. Failure to call the INCLUDE service will result in a map fault abort when a referenct is made to an address within the shared region.

Static shared memory is allocated in a similar manner for full MPX-32 as for MPX-32/M. Hence, a task designed to share memory under MPX-32/M will also execute under MPX-32 without any source changes provided all other resource requirements are compatible.

10.3 System Generation

All files associated with the generation of a MPX-32/M system reside under the user name MEMONLY. A memory-only version of MPX-32 is generated by means of a conditional assembly flag (C.MEMO) set within the MPX-32 macro library. With this flag set, five resident system source modules (SH.ALOC, SH.FISE, SH.LODR, SH.MONS and SJ.SWAPR) are reassembled to produce their corresponding memory-only object modules (under the username MEMONLY). A new OBJ file (OH.xxM) is then created using these modules for subsequent input to the SYSGEN utility.

Devices can be configured with SYSGEN via the standard CONTROLLER and DEVICE directives with the following restrictions:

(1) If more than one task wishes to allocate a specific device at activation time, that device must be configured as shared. This precludes any task from interrupting the sequential task activation process from tape of floppy disc by being placed on the resource wait queue. There is, in effect, no resource waiting during task activation. Due to this resource allocation restriction, the system console (when configured) is always treated as a shared device regardless of its SYSGEM definition.

(2) Device allocation is limited at Catalog time to ASSIGN3 (lfc to device) and ASSIGN4 (lfc to lfc). Blocked I/O is supported for magnetic tape of floppy disc, however, there is no system spooled input or output capability. Multiple tasks performing I/O to the same device must perform their own synchronization for I/O integrity.

(3) Terminal devices (other than the system console) are not supported.

Since there is no system input or output files in the memory-only environment, the system devices (/SYSDEVS) and system files (/FILES) sections need not be included in the SYSGEN directive file during system generation. With these exceptions, the SYSGEN utility is used to generate a MPX-32/M system in a manner similar to that for any user system.

10.3.1 Sample SYSGEN Directive File

```
TITLE=MPX-32/M EXAMPLE SYSTEM FOR 32/7X WITH 128KW MEMORY
//HARDWARE
/PARAMETERS
MACHINE=327X
MEMONLY
/MEMORY
SIZE=16,TYPE=E,CLASS=S
/CHANNELS
CONTROLLER=M910,PRIORITY=18,CLASS=E,HANDLER=H.MT00
DEVICE=00,DTC=M9,SHR
CONTROLLER=LP7A,PRIORITY=21,CLASS=0,HANDLER=H.LP00
DEVICE=00,DTC=LP,SHR,PHYSA=7802
CONTROLLER=CT7E,PRIORITY=23,CLASS=2,HANDLER=H.CT00
DEVICE=00,DTC=CT,SHR,PHYSA=7801,LINSIZ=80,PAGE=24
CONTROLLER=NU00
DEVICE=00,DTC=NU,SHR
/INTERRUPTS
PRIORITY=13,RTOM=(79,1C),PROGRAM=H.IP13
PRIORITY=27,RTOM=(79,18),PROGRAM=H.IP27
PRIORITY=28,RTOM=(79,06),PROGRAM=H.IPCL
PRIORITY=2C,RTOM=(79,03)
PRIORITY=7F,RTOM=(79,04),PROGRAM=H.IPIT,INTV
/TRAPS
PROGRAM=(H.IP00,H.IP01,H.IP02,H.IP03,H.IP04,H.IP05)
PROGRAM=(H.IP06,H.IP07,H.IP08,H.IP09,H.IP0E,H.IP0F)
//SOFTWARE
/PARAMETERS
SYSTEM=MEM07X
SYMTAB=MEM0TAB
DISP=20
POOL=1200
NTIM=60
MTIM=60
ITIM=384
ITLB=2C
TQFULL=1200
TQMIN=400
PATCH=8
/TABLES
SHARE=4
TIMER=4
//END
```

```
TITLE=MPX-32/M SAMPLE SYSTEM FOR 32/27 WITH 64KW MEMORY
//HARDWARE
/PARAMETERS
MACHINE=3227
MEMONLY
/MEMORY
SIZE=32,TYPE=E,CLASS=S
/CHANNELS
CONTROLLER=M910,PRIORITY=08,CLASS=F,MUX=XIO,HANDLER=H.EXIO
DEVICE=00,DTC=M9,SHR,HANDLER=H.XMT
CONTROLLER=LF7E,PRIORITY=13,CLASS=F,MUX=IOP,SUBCH=F
DEVICE=F0,DTC=FL,SHR,DISC=FL001,HANDLER=H.FLIOP
DEVICE=F1,DTC=FL,SHR,DISC=FL001,HANDLER=H.FLIOP
DEVICE=F8,DTC=LP,SHR,HANDLER=H.LPIOP
CONTROLLER=CT7E,PRIORITY=13,CLASS=F,MUX=IOP,SUBCH=F
DEVICE=FC,DTC=CT,SHR,HANDLER=H.CTIOP,IOQ=CONT,LINSIZ=80,PAGE=24
CONTROLLER=NU00
DEVICE=00,DTC=NU,SHR
/INTERRUPTS
PRIORITY=18,RTOM=(71,06),PROGRAM=H.IPCL
PRIORITY=1C,RTOM=(71,03)
PRIORITY=5F,RTOM=(71,04),PROGRAM=H.IPIT,INTV
/TRAPS
PROGRAM=(H.IP00,H.IPAS,H.IP02,H.IP03,H.IP04,H.IP05,H.IP06)
PROGRAM=(H.IP07,H.IP08,H.IP09,H.IP0A,H.IP0C,H.IP0F,H.IP13)
PROGRAM=H.IPHT
//SOFTWARE
/PARAMETERS
SYSTEM=MEM027
SYMTAB=MEMOTAB
DISP=20
POOL=1200
NTIM=60
MTIM=60
ITIM=384
ITLB=1C
TQFULL=1200
TQMIN=400
PATCH=8
/TABLES
SHARE=4
TIMER=4
//END
```

10.4 System Installation

Once a MPX-32/M system has been generated, a user system distribution medium of magnetic tape of floppy disc (SDT) is created by issuing the SDT directive to the File Manager. See Section 4.3 (this volume) for the syntax format for a memory-only system.

After creating a SDT, system installation is performed on the target machine by mounting the SDT on the appropriate device and issuing an Initial Program Load (IPL) from that device via the CPU front panel. See Section 2.5.1 (this volume) for details.

10.5 Dynamic Task Activation

Tasks can be loaded and activated from tape or floppy disc after the initial installation from a SDT. This can be accomplished by any resident task via the activate task (M.ACTV) or parameter task activation (M.PTSK) system service calls. Tasks can also be activated by issuing the ACTIVATE or ESTABLISH commands to OPCMM at the operator's console.

Upon receiving an activation request from a task other than SYSBUILD, the system allocator issued the following message at the system console:

ACTIVATION **taskname**, MOUNT MEDIA AND ENTER<CR>WHEN READY TO LOAD

When a response is given to the message, the appropriate task is loaded from the tape or floppy disc into memory and activated in a manner similar to that performed from the SDT. However, there is no intervening suspension of the activating task, and the task being loaded is immediately placed in an active state unless a wait state in indicated in the activation parameter block associated with the task. Dynamic task activation is always directed to the IPL device so the tape or floppy disc must be mounted on the same unit that was used during system installation and initialization.

The tape or floppy disc used to dynamically activate a task is created by issuing the memory-only (MEMO) directive to the File Manager. This command allows load modules to be placed on tape or floppy disc in such a way that they can be activated in the memory-only environment. The syntax of the MEMO command is:

MEMO loadmod [,loadmod] ,...

where:

loadmod — is the load module name of the task to be activated. A maximum of 20 load module names can be specified per tape or floppy disc, however, OPCMM cannot be activated in this manner.

During dynamic task activation, the system allocator reads the preamble (first 192-word block) of the first load module on the input medium. If the load module name in the preamble is the same as the load module name specified in this directive, the remainder of the load module is loaded. If the names do not match, the allocator advances down the input medium to the preamble of the next load module. This process continues until either the appropriate load module name is found or an end-of-file (written by the File Manager) is encountered. In either case, the input medium will be deallocated when task loading has completed.

10.6 Operator Communications (OPCOM)

The operator communications task (renamed OPCMM) is modified to unconditionally function as though it were activated by the system user. This causes it to be activated only when the ATTENTION key is depressed on the CPU front panel. It will not interfere with other tasks tha may be attempting to use the console when there is no input to OPCMM.

When OPCMM is encountered by SYSBUILD during system installation, the system console is momentarily specified as unshared. This allows OPCMM to be assigned as the console owner during its activation phase. The console is then specified as shared by SYSBUILD for all subsequent task activations. This allows these asks to allocate the console, if desired. In this way, all interrupts generated by depressing the ATTENTION key will be directed to OPCMM as the owner of the system console.

Several commands in the MPX-32 version of OPCOm do not apply in the MPX-32/M environment and the code associated with them is removed from OPCMM. Only the following commands are recognized by OPCMM:

	ABORT	Abort Specified Task
	ACTIVATE	Activate a Task
	BREAK	Enter Pseudo Interrupt Receiver for Specified Task
	CONNECT	Connect Task to Indirectly Connected Interrupt Level
*	CONTINUE	Release Task from HOLD
	DEBUG	Access System Debugger
	DELETETIMER	Delete Specified Timer Attached to Task
	DISABLE	Disable Interrupt at Specified Priority Level
	DISCONNECT	Disconnect Task from Indirectly Connected Interrupt
	ENABLE	Enable Interrupt at Specified Priority Level
	ENTER	Update System Date and Time
	ESTABLISH	Activate and Suspend Task
	EXIT	Remove OPCMM from System
*	HOLD	Inhibit Task from Gaining CPU Control
	KILL	Delete Specified Task from System
*	LIST	List Entries in Dispatch Queue
	MODIFY	Change Contents of Memory Word in Operating System
	OFFLINE	Make Device Unavailable for Allocation
	ONLINE	Make Device Available for Allocation
	REQUEST	Generate REQUEST-INTERRUPT for Specified Interrupt Level
	RESUME	Resume Specified Task
	SEARCH	Search Memory for Specified Value
	SETTIMER	Set Specified Timer
	SNAP	Dump Memory Location Contents to Console
	STATUS	List Memory, Device or Queue Status
	TIME	Print Time and Date at Console
	TURNON	Activate/Reactivate Task at Specified Time Interval

* These commands have reduced functionality in the memory-only environment. For details, see the individual command descriptions in the MPX-32 Reference Manual Volume 1, Chapter 4.

10.6.1 Sample Job Stream for Creation of OPCMM

```
$JOB J.OPCMM MEMONLY
$EXECUTE FILEMGR
DELETEU OJ.OPCOM
CREATEU OJ.OPCOM,DC,50
$ASSIGN1 PRE=S3227M
$ASSIGN1 SI=SJ.OPCOM
$ASSIGN1 BO=OJ.OPCOM
$ASSIGN3 UT1=DC,800,U
$OPTION 17
$ALLOCATE 25000
$EXECUTE ASSEMBLE
$ALLOCATE 18000
$ASSIGN1 SGO=OJ.OPCOM
$EXECUTE CATALOG
OPTION 25
FILES 4
BUFFERS 2
ASSIGN3 TER=CT
CATALOG OPCMM P 55
$EOJ
$$
```

10.7 System Monitor Services (H.MONS)

The system monitor services have undergone little change. All entry points are defined. The entry points that do not apply to the memory-only environment were modified to return an error condition. These entry points are:

	EP02	Permanent File Address Inquiry (SVC 1,X'43')
	EP12	Memory Dump Request (no SLO file) (SVC 1,X'4F')
	EP13	Load Overlay (SVC 1,X'50')
	EP14	Load and Execute Overlay (SVC 1,X'51')
*	EP21	Allocate File or Peripherla Device (device only) (SVC 1,X'40'0
*	EP22	Deallocate File or Device (device only) (SVC 1,X'41')
	EP27	Submit Job from Disc File (SVC 1,X'61')
	EP29	Load and Execute Interactive Debugger (SVC 1,X'63')
	EP30	Delete Interactive Debugger
	EP33	Permanent File Log (SVC 1,X'73')
	EP42	Debug Link Service (SVC 1,X'66')
*	EP55	RTM Allocate File or Device (allocates peripheral devices only; files are not supported)
	EP61	RTM Log File(s)

* These system services have reduced functionality in the memory-only environment. For details, see the individual system service descriptions for EP21 and EP 22 in the MPX-32 Reference Manual, Volume 1, Chapter 7.

In addition to the normal error conditions, the memory-only version of monitor services will issue an abort message for:

(1) An attempt to obtain a permanent file log.

(2) An attempt to load or execute the Debug overlay.

APPENDIX A
MPX-32 DEVICE ACCESS

Throughout the reference manual, the generic descriptor 'devmnc' is used to indicate 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 only, which will result in allocation of the first available device of the type requested.

A second method of device specification is achieved by using the generic device type 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, channel/controller, and device address/subaddress. This method allows specification of a particular device.

1. Special Device Specifications and Handling

1.1 Magnetic Tape

For magnetic tape, a reel identifier, multivolume number, and unblocking can be part of the device mnemonic.

Syntax:

lfc= device [,reel | ,reel,volume | ,reel,volume,U | ,reel,,U]

where:

device	is any one of the four levels of device specification described above.
reel	specifies a one- to four-character identifier for the reel. This parameter is required in batch. This parameter is not required in TSM and if not specified, the default is SCRA (Scratch).
volume	if multivolume tape, indicates volume number. Default: not multivolume (0).
U	the tape is optionally unblocked. Default: blocked.

Commas in this specification are significant. If an option is not specified, e.g., a reel identifier, but another option is specified, e.g., U, commas must be inserted for all non-specified options in between, e.g.,

MT1000,,,U

There must be no embedded blanks within the entire device mnemonic.

When the task is activated that has an assignment to tape, a MOUNT message indicates the name of the task and other information on the OPCOM console:

{TASK / jobno},taskname,taskno MOUNT reel VOL volume ON devmnc DEV,R,A,H?

where:

jobno	if the task is part of a batch job, identifies the job by job number.
taskname	is the name of the task to which the tape is assigned.
taskno	is the number of the task.

reel — if the assignment is a multivolume tape, indicates the reel identifier specified in the assignment. This parameter is required in batch. This parameter is not required in TSM and if not specified, the default is SCRA.

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.

DEV,R,A,H — the device listed in the message can be allocated and the task resumed (R), a different device can be selected (DEV), 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 will override any specification made at assignment. Values are:

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) Default.

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 then be resumed by the OPCOM CONTINUE command, at which time a tape drive will be 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 you want to use. 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 just described.

1.2 Temporary Disc File Size

For a temporary disc file, size must be specified and unblocking is optional.

Syntax:

lfc = device,size [,U]

where:

size	specifies the number of 192-word blocks required.
U	the file is optionally unblocked. Default: blocked.

Examples of the three methods of device specification follow:

Type 1 - Generic Device Class

$ASSIGN3 DEV=M9,,1

In this example, the device assigned to logical file code (lfc) "DEV" will be any 9-track tape unit on any channel. The multivolume reel number is 1. The reel identifier is SCRA.

Type 2 - Generic Device Class and Channel/Controller

$ASSIGN3 DEV=M910,MORK,,U

In this example, the device assigned to logical file code (lfc) "DEV" will be 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 supplied. This is not a multivolume tape. It is, however, unblocked.

Type 3 - Specific Device Request

$ASSIGN3 DEV=M91001

In this example, the device assigned to logical file code (lfc) "DEV" will be 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.

2. GPMC Devices

GPMC/GPDC device specifications are in keeping with the general structure just described. For instance, the terminal at subaddress 04 on GPMC 01 whose channel address is 20 would be identified as follows:

```
$ASSIGN3 DEV=TY2004
```

3. 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) upon attempt to read and normal completion upon attempt to write.

4. OPCOM Console

Logical file codes are assigned to the OPCOM console by using the device type "CT".

5. Special System Files

There are four special mnemonics provided for access to special system files: SLO, SBO, SGO and SYC. These are assigned via the $ASSIGN2 statement, as is:

```
$ASSIGN2 OUT=SLO,printlines
```

For non-batch tasks, SLO and SBO files are allocated dynamically by the system and used to disc 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.

SGO and SYC assignments are used for batch processing. See Section 7.6.

Dev Type Code	Device	Device Description
00	CT	Operator Console (Not Assignable)
01	DC	Any Disc Unit
02	DM	Any Moving Head Disc
03	DF	Any Fixed Head Disc
04	MT	Any Magnetic Tape Unit
05	M9	Any 9-Track Magnetic Tape Unit
06	M7	Any 7-Track Magnetic Tape Unit
07	CD	Any Card Reader-Punch
08	CR	Any Card Reader
09	CP	Any Card Punch
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 Disc
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)

Table A-1: Device Type Codes

6. Samples

A description of device selection possibilities would be constructed as follows:

DISC

DC	Any Disc
DM	Any Moving Head Disc
DM08	Any Moving Head Disc on Channel 08
DM0801	Moving Head Disc 01 on Channel 08
DF	Any Fixed Head Disc
DF04	Any Fixed Head Disc on Channel 04
DF0401	Fixed Head Disc 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
M7	Any 7-track Magnetic Tape
M712	Any 7-track Magnetic Tape on Channel 12
M71201	7-track Magnetic Tape 01 on Channel 12

CARD EQUIPMENT

CD	Any Card Reader-Punch
CR	Any CR
CR78	Any CR on Channel 78
CR7800	CR on Channel 78 Subaddress 00
CP	Any CP
CP7C	Any CP on Channel 7C
CP7C00	CP on Channel 7C Subaddress 00

LINE PRINTER

LP	Any LP
LP7A	Any LP on Channel 7A
LP7A00	LP on Channel 7A Subaddress 00

APPENDIX B
SYSTEM SERVICES CROSS REFERENCE CHARTS

USER LEVEL SYSTEM SERVICES - MACRO NAME

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.ACTV	ACTIVATE TASK	52	H.MONS,15	8.2.1
	M.ADRS	MEMORY ADDRESS INQUIRY	44	H.MONS,3	8.3.1
*	M.ALOC	ALLOCATE FILE OR PERIPHERAL DEVICE	40	H.MONS,21	7.8.1
	M.ANYW	WAIT FOR ANY MSG, END ACTION, OR BRK	7C	H.MONS,37	8.2.2
	M.ASYNCH	SET ASYNCHRONOUS TASK INTERRUPT	1C	H.MONS,68	8.2.3
	M.BACK	BACKSPACE RECORD	35	H.IOCS,9	7.8.2
		OR			
		FILE	36	H.IOCS,19	
	M.BORT	ABORT SPEC. TASK	56	H.MONS,19	8.2.4
		OR			
		SELF	57	H.MONS,20	
		OR			
		WITH EXT. MESSAGE	62	H.MONS,28	
	M.BRK	BREAK/TASK INTERRUPT LINK	6E	H.MONS,46	8.2.5
	M.BRKXIT	EXIT FROM TASK INTERRUPT LEVEL	70	H.MONS,48	8.2.6
**	M.CDJS	SUBMIT JOB FROM DISC FILE	61	H.MONS,27	8.2.7
	M.CLSE	CLOSE FILE	39	H.IOCS,23	7.8.3
	M.CONADB	CONVERT ASCII DECIMAL TO BINARY	28	H.TSM,7	5.6.3.1
	M.CONAHB	CONVERT ASCII HEX TO BINARY	29	H.TSM,8	5.6.3.2
	M.CONBAD	CONVERT BINARY TO ASCII DECIMAL	2A	H.TSM,9	5.6.3.3

*Reduced functionality under Memory-Only MPX-32
**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - MACRO NAME

(CONTINUED)

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.CONBAH	CONVERT BINARY TO ASCII HEX	2B	H.TSM,10	5.6.3.4
	M.CONN	CONNECT TASK TO INTERRUPT	4B	H.MONS,10	8.2.8
**	M.CREATE	CREATE PERM FILE	75	H.FISE,12	7.8.4
	M.CWAT	SYSTEM CONSOLE WAIT	3D	H.IOCS,26	7.8.5
*	M.DALC	DEALLOCATE FILE OR DEVICE	41	H.MONS,22	7.8.6
	M.DATE	DATE AND TIME INQUIRY	15	H.MONS,70	8.2.9
**	M.DEBUG	LOAD AND EXECUTE INTERACTIVE DEBUGGER	63	H.MONS,29	8.2.10
**	M.DELETE	DELETE PERM FILE	77	H.FISE,14	7.8.7
	M.DELTSK	DELETE TASK	5A	H.MONS,31	8.2.11
	M.DEVID	GET DEVICE MNEMONIC OR TYPE CODE	14	H.MONS,71	8.2.12
	M.DISCON	DISCONNECT TASK FROM INTERRUPT	5D	H.MONS,38	8.2.13
	M.DLTT	DELETE TIMER ENTRY	47	H.MONS,6	8.2.14
	M.DSMI	DISABLE MESSAGE TASK INTERRUPT	2E	H.MONS,57	8.2.15
	M.DSUB	DISABLE USER BREAK INTERRUPT	12	H.MONS,73	8.2.16
**	M.DUMP	MEMORY DUMP REQUEST	4F	H.MONS,12	8.3.2
	M.EAWAIT	END ACTION WAIT	1D	H.EXEC,40	8.2.17
	M.ENMI	ENABLE MESSAGE TASK INTERRUPT	2F	H.MONS,58	8.2.18

*Reduced functionality under Memory-Only MPX-32
**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - MACRO NAME

(CONTINUED)

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.ENUB	ENABLE USER BREAK INTERRUPT	13	H.MONS,72	8.2.19
	M.EXCL	FREE SHARED MEMORY	79	H.ALOC,14	8.3.3
	M.EXIT	TERMINATE TASK EXECUTION	55	H.MONS,18	8.2.20
**	M.FADD	PERMANENT FILE ADDRESS INQUIRY	43	H.MONS,2	7.8.8
	M.FD	FREE DYNAMIC EXTENDED INDEX DATA SPACE	6A	H.ALOC,9	8.3.5
	M.FE	FREE DYNAMIC TASK EXECUTION SPACE	68	H.ALOC,11	8.3.6
	M.FILE	OPEN FILE	30	H.IOCS,1	7.8.9
**	M.FSLR	RELEASE SYNCHRONIZATION FILE LOCK	24	H.FISE,25	7.8.10
**	M.FSLS	SET SYNCHRONIZATION FILE LOCK	23	H.FISE,24	7.8.11
	M.FWRD	ADVANCE RECORD OR FILE	33 34	H.IOCS,7 H.IOCS,8	7.8.12
	M.FXLR	RELEASE EXCLUSIVE FILE LOCK	22	H.FISE,23	7.8.13
**	M.FXLS	SET EXCLUSIVE FILE LOCK	21	H.FISE,22	7.8.14
	M.GADRL	GET ADDRESS LIMITS	65	H.MONS,41	8.3.7
	M.GD	GET DYNAMIC EXTENDED INDEXED DATA SPACE	69	H.ALOC,8	8.3.8
	M.GE	GET DYNAMIC TASK EXECUTION SPACE	67	H.ALOC,10	8.3.9

*Reduced functionality under Memory-Only MPX-32
**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - MACRO NAME

(CONTINUED)

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.GMSGP	GET MSG PARAMETERS	7A	H.MONS,35	8.2.21
	M.GRUNP	GET RUN PARAMETERS	7B	H.MONS,36	8.2.22
	M.HOLD	PROGRAM HOLD REQUEST	58	H.MONS,25	8.2.23
	M.ID	GET TASK NUMBER	64	H.MONS,32	8.2.24
*	M.INCL	GET SHARED MEMORY	72	H.ALOC,13	8.3.10
	M.INT	ACTIVATE TASK INTERRUPT	6F	H.MONS,47	8.2.25
**	M.LOG	PERMANENT FILE LOG	73	H.MONS,33	7.8.15
	M.MYID	GET TASK NUMBER	64	H.MONS,32	8.2.26
**	M.OLAY	LOAD OVERLAY	50	H.MONS,13	8.2.27
		OR			
		LOAD AND EXECUTE OVERLAY	51	H.MONS,14	
**	M.PDEV	PHYSICAL DEVICE INQUIRY	42	H.MONS,1	7.8.16
**	M.PERM	CHANGE TEMP FILE TO PERMANENT	76	H.FISE,13	7.8.17
	M.PGOW	TASK OPTION WORD INQUIRY	4C	H.MONS,24	8.2.28
	M.PRIL	CHANGE PRIORITY LEVEL	4A	H.MONS,9	8.2.29
	M.PTSK	PARAMETER TASK ACTIVATION	5F	H.MONS,40	8.2.30
	M.RCVR	RECEIVE MESSAGE LINK ADDRESS	6B	H.MONS,43	8.2.31
	M.READ	READ RECORD	31	H.IOCS,3	7.8.18
	M.RELP	RELEASE DUAL PORTED DISC	27	H.IOCS,27	7.8.19

*Reduced functionality under Memory-Only MPX-32
**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - MACRO NAME

(CONTINUED)

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.RESP	RESERVE DUAL PORTED DISC	26	H.IOCS,24	7.8.20
	M.RRES	RELEASE CHANNEL	3B	H.IOCS,13	7.8.21
	M.RSML	RESOURCEMARK LOCK	19	H.MONS,62	7.8.22
	M.RSMU	RESOURCEMARK UNLOCK	1A	H.MONS,63	7.8.23
	M.RSRV	RESERVE CHANNEL	3A	H.IOCS,12	7.8.24
	M.RWND	REWIND FILE	37	H.IOCS,2	7.8.25
	M.SETS	SET USER STATUS WORD	48	H.MONS,7	8.2.32
	M.SETT	CREATE TIMER ENTRY	45	H.MONS,4	8.2.33
*	M.SHARE	SHARE MEMORY WITH ANOTHER TASK	71	H.ALOC,12	8.3.11
	M.SMSGR	SEND MESSAGE TO SPECIFIED TASK	6C	H.MONS,44	8.2.34
	M.SMULK	UNLOCK AND DEQUEUE SHARED MEMORY	1F	H.ALOC,19	8.3.12
	M.SRUNR	SEND RUN REQUEST	6D	H.MONS,45	8.2.35
	M.SUAR	SET USER ABORT RECEIVER ADDRESS	60	H.MONS,26	8.2.36
	M.SUME	RESUME TASK EXECUTION	53	H.MONS,16	8.2.37
	M.SUSP	SUSPEND TASK EXECUTION	54	H.MONS,17	8.2.38
	M.SYNCH	SET SYNCHRONOUS TASK INTERRUPT	1B	H.MONS,67	8.2.39
	M.TBRKON	TRAP ONLINE USER'S TASK	5C	H.TSM,6	5.6.2

*Reduced functionality under Memory-Only MPX-32
**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - MACRO NAME

(CONTINUED)

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.TDAY	TIME-OF-DAY INQUIRY	4E	H.MONS,11	8.2.40
**	M.TSCAN	SCAN TERMINAL INPUT BUFFER	5B	H.TSM,2	5.6.1
	M.TSTE	ARITHMETIC EXCEPTION INQUIRY	4D	H.MONS,23	8.2.41
	M.TSTS	TEST USER STATUS WORD	49	H.MONS,8	8.2.42
	M.TSTT	TEST TIMER ENTRY	46	H.MONS,5	8.2.43
	M.TURNON	ACTIVATE PROGRAM AT GIVEN TIME OF DAY	1E	H.MONS,66	8.2.44
	M.TYPE	CONSOLE TYPE	3F	H.IOCS,14	7.8.26
	M.UPSP	UPSPACE	10	H.IOCS,20	7.8.27
	M.USER	USERNAME SPECIFICATION	74	H.MONS,34	7.8.28
	M.WAIT	WAIT I/O	3C	H.IOCS,25	7.8.29
	M.WEOF	WRITE EOF	38	H.IOCS,5	7.8.30
	M.WRIT	WRITE RECORD	32	H.IOCS,4	7.8.31
	M.XBRKR	EXIT FROM TASK INTERRUPT LEVEL	70	H.MONS,48	8.2.45
	M.XIEA	NO-WAIT I/O END ACTION RETURN	2C	H.IOCS,34	7.8.32
	M.XMEA	EXIT FROM MESSAGE END ACTION ROUTINE	7E	H.MONS,50	8.2.46
	M.XMSGR	EXIT FROM MESSAGE RECEIVER	5E	H.MONS,39	8.2.47
	M.XREA	EXIT RUN REQUEST END ACTION ROUTINE	7F	H.MONS,51	8.2.48

**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - MACRO NAME

(CONTINUED)

	MACRO	DESCRIPTOR	SVC 1,XX	MODULE, E.P.	REF MANUAL SECTION
	M.XRUNR	EXIT RUN RECEIVER	7D	H.MONS,49	8.2.49
	M.XTIME	TASK CPU EXECUTION TIME	2D	H.MONS,65	8.2.50
	N/A	ERASE OR PUNCH TRAILER	3E	H.IOCS,21	7.8.33
**	N/A	DEBUG LINK SERVICE	66	H.MONS,42	8.2.51
	N/A	EXECUTE CHANNEL PROGRAM	25	H.IOCS,10	7.8.34
	N/A	RELEASE FHD PORT	27	H.IOCS,27	7.8.35
	N/A	RESERVE FHD PORT	26	H.IOCS,24	7.8.36
	N/A	SET TABS IN UDT	59	H.TSM,5	N/A

**Not supported under Memory-Only MPX-32

USER LEVEL SYSTEM SERVICES - ALPHABETIC

		SVC 1,XX	MODULE	E.P.	REF MANUAL SECTION
ABORT SELF	M.BORT	57	H.MONS	20	8.2.4
ABORT SPECIFIED TASK	M.BORT	56	H.MONS	19	8.2.4
ABORT WITH EXTENDED MESSAGE	M.BORT	62	H.MONS	28	8.2.4
ACTIVATE PROGRAM AT GIVEN TIME OF DAY	M.TURNON	1E	H.MONS	66	8.2.44
ACTIVATE TASK INTERRUPT	M.INT	6F	H.MONS	47	8.2.25
ACTIVATE TASK	M.ACTV	52	H.MONS	15	8.2.1
ADVANCE FILE OR RECORD	M.FWRD	34 33	H.IOCS H.IOCS	8 7	7.8.12 7.8.12
ALLOCATE FILE OR PERIPHERAL DEVICE	M.ALOC	40	H.MONS	21	7.8.1
ARITHMETIC EXCEPTION INQUIRY	M.TSTE	4D	H.MONS	23	8.2.41
BACKSPACE FILE OR RECORD	M.BACK	36 35	H.IOCS H.IOCS	19 9	7.8.2 7.8.2
BREAK/TASK INTERRUPT LINK	M.BRK	6E	H.MONS	46	8.2.5
CHANGE PRIORITY LEVEL (PRIV)	M.PRIL	4A	H.MONS	9	8.2.29
CHANGE TEMP FILE TO PERMANENT	M.PERM	76	H.FISE	13	7.8.17
CLOSE FILE	M.CLSE	39	H.IOCS	23	7.8.3
CONNECT TASK TO INTERRUPT	M.CONN	4B	H.MONS	10	8.2.8
CONSOLE TYPE	M.TYPE	3F	H.IOCS	14	7.8.26
CONVERT ASCII DECIMAL TO BINARY	M.CONADB	28	H.TSM	7	5.6.3.1

UNPRIVILEGED USER LEVEL SYSTEM SERVICES - ALPHABETIC

(CONTINUED)

		SVC 1,XX	MODULE	E.P.	REF MANUAL SECTION
CONVERT ASCII HEX TO BINARY	M.CONAHB	29	H.TSM	8	5.6.3.2
CONVERT BINARY TO ASCII DECIMAL	M.CONBAD	2A	H.TSM	9	5.6.3.3
CONVERT BINARY TO ASCII HEX	M.CONBAH	2B	H.TSM	10	5.6.3.4
CREATE PERM FILE	M.CREATE	75	H.FISE	12	7.8.4
CREATE TIMER ENTRY	M.SETT	45	H.MONS	4	8.2.33
*DATE AND TIME INQUIRY	M.DATE	15	H.MONS	70	8.2.9
DEALLOCATE FILE OR DEVICE	M.DALC	41	H.MONS	22	7.8.6
DEBUG LINK SERVICE	N/A	66	H.MONS	42	8.2.51
DELETE PERM FILE	M.DELETE	77	H.FISE	14	7.8.7
DELETE TASK	M.DELTSK	5A	H.MONS	31	8.2.11
DELETE TIMER ENTRY	M.DLTT	47	H.MONS	6	8.2.14
DISABLE MESSAGE TASK INTERRUPT	M.DSMI	2E	H.MONS	57	8.2.15
*DISABLE USER BREAK INTERRUPT	M.DSUB	12	H.MONS	73	8.2.16
DISCONNECT TASK FROM INTERRUPT	M.DISCON	5D	H.MONS	38	8.2.13
ENABLE MESSAGE TASK INTERRUPT	M.ENMI	2F	H.MONS	58	8.2.18
*ENABLE USER BREAK INTERRUPT	M.ENUB	13	H.MONS	13	8.2.19
END ACTION WAIT	M.EAWAIT	1D	H.EXEC	40	8.2.17
ERASE OR PUNCH TRAILER	N/A	3E	H.IOCS	21	7.8.33

*NEW

UNPRIVILEGED USER LEVEL SYSTEM SERVICES - ALPHABETIC

(CONTINUED)

		SVC 1,XX	MODULE	E.P.	REF MANUAL SECTION
EXECUTE CHANNEL PROGRAM	N/A	25	H.IOCS	10	7.8.34
EXIT FROM MSG RCVR	M.XMSGR	5E	H.MONS	39	8.2.47
EXIT FROM MSG END ACTION ROUTINE	M.XMEA	7E	H.MONS	50	8.2.46
EXIT FROM TASK INTERRUPT LEVEL	M.BRKXIT M.XBRKR	70	H.MONS	48	8.2.6 8.2.45
EXIT RUN RCVR	M.XRUNR	7D	H.MONS	49	8.2.49
EXIT FROM RUN REQUEST END ACTION ROUTINE	M.XREA	7F	H.MONS	51	8.2.48
FREE DYNAMIC EXT INDEX DATA SPACE	M.FD	6A	H.ALOC	9	8.3.5
FREE DYNAMIC TASK EXECUTION SPACE	M.FE	68	H.ALOC	11	8.3.6
FREE SHARED MEMORY	M.EXCL	79	H.ALOC	14	8.3.3
GET ADDRESS LIMITS	M.GADRL	65	H.MONS	41	8.3.7
*GET DEVICE MNEMONIC OR TYPE CODE	M.DEVID	14	H.MONS	71	8.2.12
GET DYNAMIC EXTENDED INDEXED DATA SPACE	M.GD	69	H.ALOC	8	8.3.8
GET DYNAMIC TASK EXECUTION SPACE	M.GE	67	H.ALOC	10	8.3.9
GET MSG PARAMETERS	M.GMSGP	7A	H.MONS	35	8.2.21
GET RUN PARAMETERS	M.GRUNP	7B	H.MONS	36	8.2.22
GET SHARED MEMORY	M.INCL	72	H.ALOC	13	8.3.10
GET TASK NUMBER	M.ID M.MYID	64	H.MONS	32	8.2.24 8.2.26

*NEW

UNPRIVILEGED USER LEVEL SYSTEM SERVICES - ALPHABETIC

(CONTINUED)

		SVC 1,XX	MODULE	E.P.	REF MANUAL SECTION
LOAD AND EXECUTE INTERACTIVE DEBUGGER	M.DEBUG	63	H.MONS	29	8.2.10
LOAD OVERLAY AND LOAD AND EXECUTE OVERLAY	M.OLAY	5O 51	H.MONS H.MONS	13 14	8.2.27 8.2.27
MEMORY ADDRESS INQUIRY	M.ADRS	44	H.MONS	3	8.3.1
MEMORY DUMP REQUEST	M.DUMP	4F	H.MONS	12	8.3.2
NO-WAIT I/O END ACTION RETURN	M.XIEA	2C	H.IOCS	34	7.8.32
OPEN FILE	M.FILE	30	H.IOCS	1	7.8.9
PARAMETER TASK ACTIVATION (PRIV)	M.PTSK	5F	H.MONS	40	8.2.30
PERMANENT FILE ADDRESS INQUIRY	M.FADD	43	H.MONS	2	7.8.8
PERMANENT FILE LOG	M.LOG	73	H.MONS	33	7.8.15
PHYSICAL DEVICE INQUIRY	M.PDEV	42	H.MONS	1	7.8.16
PROGRAM HOLD REQUEST	M.HOLD	58	H.MONS	25	8.2.23
READ RECORD	M.READ	31	H.IOCS	3	7.8.18
RECEIVE MESSAGE LINK ADDRESS	M.RCVR	6B	H.MONS	43	8.2.31
RELEASE CHANNEL (PRIV)	M.RRES	3B	H.IOCS	13	7.8.21
RELEASE DUAL PORTED DISC	M.RELP	27	H.IOCS	27	7.8.19
RELEASE EXCLUSIVE FILE LOCK	M.FXLR	22	H.FISE	23	7.8.13
RELEASE FHD PORT (PRIV)	N/A	27	H.IOCS	27	7.8.35

*NEW

UNPRIVILEGED USER LEVEL SYSTEM SERVICES - ALPHABETIC

(CONTINUED)

		SVC 1,XX	MODULE	E.P.	REF MANUAL SECTION
RELEASE SYNCHRONIZATION FILE LOCK	M.FSLR	24	H.FISE	25	7.8.10
RESERVE CHANNEL (PRIV)	M.RSRV	3A	H.IOCS	12	7.8.24
*RESERVE DUAL PORTED DISC	M.RESP	26	H.IOCS	24	7.8.20
RESERVE FHD PORT (PRIV)	N/A	26	H.IOCS	24	7.8.36
RESOURCEMARK LOCK	M.RSML	19	H.MONS	62	7.8.22
RESOURCEMARK UNLOCK	M.RSMU	1A	H.MONS	63	7.8.23
RESUME TASK EXECUTION	M.SUME	53	H.MONS	16	8.2.37
REWIND FILE	M.RWND	37	H.IOCS	2	7.8.25
SCAN TERMINAL INPUT BUFFER	M.TSCAN	5B	H.TSM	2	5.6.1
SEND MESSAGE TO SPECIFIED TASK	M.SMSGR	6C	H.MONS	44	8.2.34
SEND RUN REQUEST	M.SRUNR	6D	H.MONS	45	8.2.35
SET ASYNCHRONOUS TASK INTERRUPT	M.ASYNCH	1C	H.MONS	68	8.2.3
SET EXCLUSIVE FILE LOCK	M.FXLS	21	H.FISE	22	7.8.14
SET SYNCHRONIZATION FILE LOCK	M.FSLS	23	H.FISE	24	7.8.11
SET SYNCHRONOUS TASK INTERRUPT	M.SYNCH	1B	H.MONS	67	8.2.39
SET TABS IN UDT	N/A	59	H.TSM	5	N/A
SET USER STATUS WORD	M.SETS	48	H.MONS	7	8.2.32
SET USER ABORT RECEIVER ADDRESS	M.SUAR	60	H.MONS	26	8.2.36

*NEW

UNPRIVILEGED USER LEVEL SYSTEM SERVICES - ALPHABETIC

(CONTINUED)

		SVC 1,XX	MODULE	E.P.	REF MANUAL SECTION
SHARE MEMORY WITH ANOTHER TASK	M.SHARE	71	H.ALOC	12	8.3.11
SUBMIT JOB FROM DISC FILE	M.CDJS	61	H.MONS	27	8.2.7
SUSPEND TASK EXECUTION	M.SUSP	54	H.MONS	17	8.2.38
SYSTEM CONSOLE WAIT	M.CWAT	3D	H.IOCS	26	7.8.5
TASK CPU EXECUTION TIME	M.XTIME	2D	H.MONS	65	8.2.50
TASK OPTION WORD INQUIRY	M.PGOW	4C	H.MONS	24	8.2.28
TERMINATE TASK EXECUTION	M.EXIT	55	H.MONS	18	8.2.20
TEST TIMER ENTRY	M.TSTT	46	H.MONS	5	8.2.43
TEST USER STATUS WORD	M.TSTS	49	H.MONS	8	8.2.42
TIME-OF-DAY INQUIRY	M.TDAY	4E	H.MONS	11	8.2.40
TRAP ONLINE USER'S TASK	M.TBRKON	5C	H.TSM	6	5.6.2
UNLOCK AND DEQUEUE SHARED MEMORY	M.SMULK	1F	H.ALOC	19	8.3.12
*UPSPACE	M.UPSP	10	H.IOCS	20	7.8.27
USERNAME SPECIFICATION	M.USER	74	H.MONS	34	7.8.28
WAIT I/O	M.WAIT	3C	H.IOCS	25	7.8.29
WAIT FOR ANY MSG, END ACTION, OR BRK	M.ANYW	7C	H.MONS	37	8.2.2
WRITE EOF	M.WEOF	38	H.IOCS	5	7.8.30
WRITE RECORD	M.WRIT	32	H.IOCS	4	7.8.31

*NEW

USER LEVEL SYSTEM SERVICES - SVC ORDER

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
00-0F	RESERVED			
10	UPSPACE	H.IOCS,20	M.UPSP	7.8.27
11	RESERVED			
12	DISABLE USER BREAK INTERRUPT	H.MONS,73	M.DSUB	8.2.16
13	ENABLE USER BREAK INTERRUPT	H.MONS,72	M.ENUB	8.2.19
14	GET DEVICE MNEMONIC OR TYPE CODE	H.MONS,71	M.DEVID	8.2.12
15	DATE AND TIME INQUIRY	H.MONS,70	M.DATE	8.2.9
16	RESERVED			
17	ADI I/O	H.ADIO,8	N/A	N/A
18	ADI EAI	H.ADIO,9	N/A	N/A
19	RESOURCEMARK LOCK	H.MONS,62	M.RSML	7.8.22
1A	RESOURCEMARK UNLOCK	H.MONS,63	M.RSMU	7.8.23
1B	SET SYNCHRONOUS TASK INTERRUPT	H.MONS,67	M.SYNCH	8.2.39
1C	SET ASYNCHRONOUS TASK INTERRUPT	H.MONS,68	M.ASYNCH	8.2.3
1D	END ACTION WAIT	H.EXEC,40	M.EAWAIT	8.2.17
1E	ACTIVATE PROGRAM AT GIVEN TIME OF DAY	H.MONS,66	M.TURNON	8.2.44
1F	UNLOCK AND DEQUEUE SHARED MEMORY	H.ALOC,19	M.SMULK	8.3.12
20	RESERVED			
21	SET EXCLUSIVE FILE LOCK	H.FISE,22	M.FXLS	7.8.14

USER LEVEL SYSTEM SERVICES - SVC ORDER (CONTINUED)

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
22	RELEASE EXCLUSIVE FILE LOCK	H.FISE,23	M.FXLR	7.8.13
23	SET SYNCHRONIZATION FILE LOCK	H.FISE,24	M.FSLS	7.8.11
24	RELEASE SYNCHRONIZATION FILE LOCK	H.FISE,25	M.FSLR	7.8.10
25	EXECUTE CHANNEL PROGRAM	H.IOCS,10	N/A	7.8.34
26	RESERVE FHD PORT RESERVE DUAL PORTED DISC	H.IOCS,24 H.IOCS,24	N/A M.RESP	7.8.36 7.8.20
27	RELEASE FHD PORT RELEASE DUAL PORTED DISC	H.IOCS,27 H.IOCS,27	N/A M.RELP	7.8.35 7.8.19
28	CONVERT ASCII DECIMAL TO BINARY	H.TSM,7	M.CONADB	5.6.3.1
29	CONVERT ASCII HEX TO BINARY	H.TSM,8	M.CONAHB	5.6.3.2
2A	CONVERT BINARY TO ASCII DECIMAL	H.TSM,9	M.CONBAD	5.6.3.3
2B	CONVERT BINARY TO ASCII HEX	H.TSM,10	M.CONBAH	5.6.3.4
2C	NO-WAIT I/O END ACTION RETURN	H.IOCS,34	M.XIEA	7.8.32
2D	TASK CPU EXECUTION TIME	H.MONS,65	M.XTIME	8.2.50
2E	DISABLE MESSAGE TASK INTERRUPT	H.MONS,57	M.DSMI	8.2.15

USER LEVEL SYSTEM SERVICES - SVC ORDER

(CONTINUED)

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
2F	ENABLE MESSAGE TASK INTERRUPT	H.MONS,58	M.ENMI	8.2.18
30	OPEN FILE	H.IOCS,1	M.FILE	7.8.9
31	READ RECORD	H.IOCS,3	M.READ	7.8.18
32	WRITE RECORD	H.IOCS,4	M.WRIT	7.8.31
33	ADVANCE RECORD	H.IOCS,7	M.FWRD	7.8.12
34	ADVANCE FILE	H.IOCS,8	M.FWRD	7.8.12
35	BACKSPACE RECORD	H.IOCS,9	M.BACK	7.8.2
36	BACKSPACE FILE	H.IOCS,19	M.BACK	7.8.2
37	REWIND FILE	H.IOCS,2	M.RWND	7.8.25
38	WRITE EOF	H.IOCS,5	M.WEOF	7.8.30
39	CLOSE FILE	H.IOCS,23	M.CLSE	7.8.3
3A	RESERVE CHANNEL (PRIV)	H.IOCS,12	M.RSRV	7.8.24
3B	RELEASE CHANNEL (PRIV)	H.IOCS,13	M.RRES	7.8.21
3C	WAIT I/O	H.IOCS,25	M.WAIT	7.8.29
3D	SYSTEM CONSOLE WAIT	H.IOCS,26	M.CWAT	7.8.5
3E	ERASE OR PUNCH TRAILER	H.IOCS,21	N/A	7.8.33
3F	CONSOLE TYPE	H.IOCS,14	M.TYPE	7.8.26
40	ALLOCATE FILE OR PERIPHERAL DEVICE	H.MONS,21	M.ALOC	7.8.1
41	DEALLOCATE FILE OR DEVICE	H.MONS,22	M.DALC	7.8.6
42	PHYSICAL DEVICE INQUIRY	H.MONS,1	M.PDEV	7.8.16

USER LEVEL SYSTEM SERVICES - SVC ORDER

(CONTINUED)

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
43	PERMANENT FILE ADDRESS INQUIRY	H.MONS,2	M.FADD	7.8.8
44	MEMORY ADDRESS INQUIRY	H.MONS,3	M.ADRS	8.3.1
45	CREATE TIMER ENTRY	H.MONS,4	M.SETT	8.2.33
46	TEST TIMER ENTRY	H.MONS,5	M.TSTT	8.2.43
47	DELETE TIMER ENTRY	H.MONS,6	M.DLTT	8.2.14
48	SET USER STATUS WORD	H.MONS,7	M.SETS	8.2.32
49	TEST USER STATUS WORD	H.MONS,8	M.TSTS	8.2.42
4A	CHANGE PRIORITY LEVEL (PRIV)	H.MONS,9	M.PRIL	8.2.29
4B	CONNECT TASK TO INTERRUPT	H.MONS,10	M.CONN	8.2.8
4C	TASK OPTION WORD INQUIRY	H.MONS,24	M.PGOW	8.2.28
4D	ARITHMETIC EXCEPTION INQUIRY	H.MONS,23	M.TSTE	8.2.41
4E	TIME-OF-DAY INQUIRY	H.MONS,11	M.TDAY	8.2.40
4F	MEMORY DUMP REQUEST	H.MONS,12	M.DUMP	8.3.2
50	LOAD OVERLAY	H.MONS,13	M.OLAY	8.2.27
51	LOAD AND EXECUTE OVERLAY	H.MONS,14	M.OLAY	8.2.27
52	ACTIVATE TASK	H.MONS,15	M.ACTV	8.2.1
53	RESUME TASK EXECUTION	H.MONS,16	M.SUME	8.2.37

USER LEVEL SYSTEM SERVICES - SVC ORDER

(CONTINUED)

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
54	SUSPEND TASK EXECUTION	H.MONS,17	M.SUSP	8.2.38
55	TERMINATE TASK EXECUTION	H.MONS,18	M.EXIT	8.2.20
56	ABORT SPECIFIED TASK	H.MONS,19	M.BORT	8.2.4
57	ABORT SELF	H.MONS,20	M.BORT	8.2.4
58	PROGRAM HOLD REQUEST	H.MONS,25	M.HOLD	8.2.23
59	SET TABS IN UDT	H.TSM,5	N/A	N/A
5A	DELETE TASK	H.MONS,31	M.DELTSK	8.2.11
5B	SCAN TERMINAL INPUT BUFFER	H.TSM,2	M.TSCAN	5.6.1
5C	TRAP ONLINE USER'S TASK	H.TSM,6	M.TBRKON	5.6.2
5D	DISCONNECT TASK FROM INTERRUPT	H.MONS,38	M.DISCON	8.2.13
5E	EXIT FROM MESSAGE RECEIVER	H.MONS,39	M.XMSGR	8.2.47
5F	PARAMETER TASK ACTIVATION (PRIV)	H.MONS,40	M.PTSK	8.2.30
60	SET USER ABORT RECEIVER ADDRESS	H.MONS,26	M.SUAR	8.2.36
61	SUBMIT JOB FROM DISC FILE	H.MONS,27	M.CDJS	8.2.7
62	ABORT WITH EXTENDED MESSAGE	H.MONS,28	M.BORT	8.2.4

USER LEVEL SYSTEM SERVICES - SVC ORDER

(CONTINUED)

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
63	LOAD AND EXECUTE INTERACTIVE DEBUGGER	H.MONS,29	M.DEBUG	8.2.10
64	GET TASK NUMBER	H.MONS,32	M.ID M.YID	8.2.24 8.2.26
65	GET ADDRESS LIMITS	H.MONS,41	M.GADRL	8.3.7
66	DEBUG LINK SERVICE	H.MONS,42	N/A	8.2.51
67	GET DYNAMIC TASK EXECUTION SPACE	H.ALOC,10	M.GE	8.3.9
68	FREE DYNAMIC TASK EXECUTION SPACE	H.ALOC,11	M.FE	8.3.6
69	GET DYNAMIC EXTENDED INDEXED DATA SPACE	H.ALOC,8	M.GD	8.3.8
6A	FREE DYNAMIC EXTENDED INDEX DATA SPACE	H.ALOC,9	M.FD	8.3.5
6B	RECEIVE MESSAGE LINK ADDRESS	H.MONS,43	M.RCVR	8.2.31
6C	SEND MESSAGE TO SPECIFIED TASK	H.MONS,44	M.SMSGR	8.2.34
6D	SEND RUN REQUEST	H.MONS,45	M.SRUNR	8.2.35
6E	BREAK/TASK INTERRUPT LINK	H.MONS,46	M.BRK	8.2.5
6F	ACTIVATE TASK INTERRUPT	H.MONS,47	M.INT	8.2.25
70	EXIT FROM TASK INTERRUPT LEVEL	H.MONS,48	M.BRKXIT M.XBRKR	8.2.6 8.2.45
71	SHARE MEMORY WITH ANOTHER TASK	H.ALOC,12	M.SHARE	8.3.11
72	GET SHARED MEMORY	H.ALOC,13	M.INCL	8.3.10
73	PERMANENT FILE LOG	H.MONS,33	M.LOG	7.8.15

USER LEVEL SYSTEM SERVICES - SVC ORDER

(CONTINUED)

SVC 1	DESCRIPTION	MODULE,EP	MACRO	REF MANUAL SECTION
74	USERNAME SPECIFICATION	H.MONS,34	M.USER	7.8.28
75	CREATE PERM FILE	H.FISE,12	M.CREATE	7.8.4
76	CHANGE TEMP FILE TO PERMANENT	H.FISE,13	M.PERM	7.8.17
77	DELETE PERM FILE	H.FISE,14	M.DELETE	7.8.7
78	RESERVED FOR FUTURE USE			
79	FREE SHARED MEMORY	H.ALOC,14	M.EXCL	8.3.3
7A	GET MSG PARAMETERS	H.MONS,35	M.GMSGP	8.2.21
7B	GET RUN PARAMETERS	H.MONS,36	M.GRUNP	8.2.22
7C	WAIT FOR ANY MSG, END ACTION, OR BRK	H.MONS,37	M.ANYW	8.2.2
7D	EXIT RUN RECEIVER	H.MONS,49	M.XRUNR	8.2.49
7E	EXIT FROM MESSAGE END ACTION ROUTINE	H.MONS,50	M.XMEA	8.2.46
7F	EXIT FROM RUN REQUEST END ACTION ROUTINE	H.MONS,51	M.XREA	8.2.28

APPENDIX C

MPX-32 ABORT AND CRASH CODES

Address Specification Trap Handler (H.IP0C)

CODE	DESCRIPTION
AD01	Address specification occurred within the operating system.
AD02	Address specification occurred within the current task.
AD03	Trap occurred while no tasks were in active state.
AD04	Trap occurred within another interrupt trap routine.

CODE	DESCRIPTION
AL01-AL06	Reserved
AL07	The combined number of 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	Reserved
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	Reserved
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 either not contained in the user name file M.KEY or a correct user key is not present. Also: task has attempted to deallocate TSA.

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 has already been allocated by caller (e.g., a card reader may already 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 SMD.
AL35	Attempt to share an SMD 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 on SMD.
AL41	SMD definition not a dynamic definition.
AL42	Invalid password for this 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-AL54	Reserved
AL55	The sum of the CSECT, DSECT, and the operating system sizes is greater than the total amount of memory configured.
AL56	Unrecoverable I/O error to the SMD.
AL57	File Lock Table (FLT) is full.

CODE	DESCRIPTION
AS01	Physical end-of-file encountered on write to the General Object (GO) file.
AS02	Physical end-of-file encountered on write to the Binary Output (BO) file.
AS03	Physical end-of-file encountered on write to the Listed Output (LO) file.
AS04	Physical end-of-file encountered on write to the scratch (UT1) file (i.e., $ASSIGN3 UT1 = DC, ????).
AS05	Physical end-of-file encountered on write to the cross-reference (UT2) file (i.e., $ASSIGN3 UT2 = DC,????).
AS06	There does not exist a prime number of three-word entries in the allocated core for the symbol table.
AS07	Unrecoverable I/O error on the Binary Output (BO) file.
AS08	Unrecoverable I/O error on the General Object (GO) file.
AS09	Unrecoverable I/O error on the Listed Output (LO) file.
AS10	Unrecoverable I/O error on the Source Input (SI) file.
AS11	Unrecoverable I/O error on the intermediate compressed source (UT1) file.
AS12	Physical end-of-file encountered on write to the Compressed Source Output (CS) file.
AS13	Checksum error on compressed source input either during pass 1 while reading compressed source from the Source Input (SI) file or during pass 2 while reading the intermediate scratch compressed source (UT1) file.
AS14	The file the Assembler is using as the macro library was not successfully created by the macro library generator. The file is invalid.
AS15	Unrecoverable I/O error on the Macro Library (MAC) file.
AS16	Unrecoverable I/O error on the cross-reference (UT2) file.
AS17	Unrecoverable I/O error on the Compressed Source Output (CS) file.

AS18	Invalid blocking buffer control pointer encountered on the Binary Output (BO) file.
AS19	Invalid blocking buffer control pointer encountered on the General Object (GO) file.
AS20	Invalid blocking buffer control pointer encountered on the Listed Output (LO) file.
AS21	Invalid blocking buffer control pointer encountered on the Source Input (SI) file.
AS22	Invalid blocking buffer control pointer encountered on the Scratch Compressed Source (UT1) file.
AS23	Invalid blocking buffer control pointer encountered on the Compressed Source Output (CS) file.
AS24	Invalid blocking buffer control pointer encountered on the cross-reference (UT2) file.
AS25	The macro library (MAC) file is unblocked.
AS26	End of file on MA2 file.
AS27	Unrecoverable I/O error on MA2 file.
AS28	Invalid blocking buffer control pointer on MA2 file.
AS29	MAC assigned to illegal device.
AS30	MA2 assigned to illegal device.
AS31	Potential abort conditions have been detected during program assembly. An abort status flag within the job's TSA has been set during assembler termination processing. Conditional job control directives may be used to test status prior to job continuation.
AS32	Unrecoverable I/O error on the prefix (MPXPRE) file.
AS33	Invalid blocking buffer control pointer encountered on the prefix (MPXPRE) file.

Auto-Start Trap Processor (H.IPAS)

CODE	DESCRIPTION
AU01	Trap occurred on auto-start.
AU02	Trap occurred in another interrupt trap routine.
AU03	Reserved
AU04	Reserved
AU05	User was unmapped when trap occurred.

Debugger

CODE	DESCRIPTION
DB01	End of medium error on lfc #OT in batch mode.
DB02	Fatal I/O error on lfc contained in the abort message.

Call Monitor Interrupt Processor (H.IP27 and H.IP0A)

CODE	DESCRIPTION
CM01	Physical end-of-file encountered on write to the compressed source output (CS) file. Call monitor interrupt processor cannot locate the CALM instruction.
CM02	Expected CALM instruction does not have CALM (X'30') opcode.
CMO3	Invalid CALM number.
CM04	CALM number too low (out of bounds).
CM05	CALM number too big (out of bounds).

CODE	DESCRIPTION
CT01	Physical end-of-file encountered on subroutine library. The lfc of the library in question is displayed. This results from the library being updated by another user while it is allocated by the Cataloger.
CT02	Load module file specified with CATALOG cannot be allocated.
CT03	Unrecoverable I/O error encountered on the DATAPOOL dictionary file assigned to DPD.
CT04	Listed output space is deleted and additional SLO space cannot be allocated.
CT05	Unrecoverable I/O error on file or device assigned to SBM for SYMTAB output.
CT06	An error occurred during the cataloging process and the reason is described in the SLO.

CODE	DESCRIPTION
DP01	Unrecoverable I/O error while reading or writing the DATAPOOL dictionary.
DP02	Dictionary file code (DPD) unassigned.
DP03	Unrecoverable error on error audit trail (ER) file.
DP04	Unrecoverable error on audit trail (LO) file.
DP05	Unable to allocate additional SLO space for audit trail after initial file is filled.
DP06	Unable to allocate additional SLO space for error audit trail after initial file is filled.
DP07	Invalid directive.
DP08	The name 'DATAPOOL' is not defined as a core partition.
DP09	Dictionary overflow.
DP10	Unable to reassign the DPD file.
DP11	End-of-tape or illegal end-of-file encountered on IN.
DP12	Physical end-of-media encountered on OT.
DP13	Unrecoverable error on IN.
DP14	Unrecoverable error on OT.
DP15	File code OT unassigned and the SAVE function requested.
DP16	File code IN unassigned and the REMAP function requested.
DP17	Sequence error on dictionary entry record (accessed through file code IN).
DP18	Checksum error on dictionary entry record (accessed through file code OT).
DP19	Invalid specification on REMAP directive.
DP20	Invalid specification on DPD directive.
DP21	Unrecoverable error on directive input (SYC) file.
DP22	Dictionary size is less than the required minimum (five records).
DP99	A non-fatal error occurred.

Error Condition Codes for DPEDIT

CODE	DESCRIPTION
EC11	Attempt to delete a symbol not found in the dictionary.
EC12	Attempt to delete a symbol that is used as a base for another variable.
EC13	A change is requested for a symbol used as a base that may result in a change in the relative address.
EC14	The calculated relative address does not fall on the specified boundary (precision).
EC15	The referenced base symbol is not in the datapool dictionary.
EC16	Attempt to add a symbol that is already defined in the dictionary.
EC17	The calculated relative address is not within the range of the datapool core partition.
EC18	The datapool variable does not reside in the dictionary at the location computed by the hash coding scheme.
EC19	Invalid specification on directive.
EC20	Log function deleted, not enough memory to sort data.
EC21	Log function deleted, the scratch sort file is not enough to contain the necessary data.
EC22	Log function deleted, unrecoverable I/O error on the scratch sort file.
EC23	Attempted to change a symbol not found in the dictionary.
EC24	Computed relative address does not agree with actual.
EC25	Entries are multiply defined.
ERnn	Error encountered in processing data card fields. The column number in which the error was detected is specified by "nn".

CODE	DESCRIPTION
ED01	User terminal I/O hardware error.
ED02	Internal line linkage invalid.
ED03	Reserved (RTM Only).
ED04	Internal logic error.

CODE	DESCRIPTION
FM01	Invalid command verb on directive.
FM02	Required argument(s) absent from a directive.
FM03	Create requested for an existing file.
FM04	Device specified is invalid for this command.
FM05	Decimal number specification contains non-decimal digits.
FM06	Hexadecimal number specification contains non-hexadecimal characters.
FM07	Specified file is password protected and correct password not specified.
FM08	Attempt to expand a core partition.
FM09	Cannot create or expand file due to unavailability of disc space for allocation.
FM10	Attempt to create a fast file which mapped into an existing fast file in the SMD.
FM11	DELETE, SAVE, or EXPAND requested for a disc file which does not exist.
FM12	Insufficient file assignment table space for I/O to the SMD.
FM13	Unrecoverable I/O error to the SMD.
FM14	Unrecoverable I/O error to the SYC file.
FM15	Unrecoverable I/O error to the SLO file.
FM16	Unrecoverable I/O error to the IN file.
FM17	Unrecoverable I/O error to the OUT file.
FM18	Invalid argument.
FM19	Cannot allocate required file.
FM20	Unrecoverable I/O error on SAVE/RESTORE file.
FM21	Unexpected EOF on IN file.
FM24	File specified for RESTORE not found.
FM25	Too many prototypes specified.

FM26	EOF expected on IN file not found.
FM41	End of medium on lfc SLO.
FM42	Invalid username or key.
FM99	Directive errors have been detected during execution of the File Manager. An abort status flag within the job's TSA has been set during File Manager termination processing. Conditional job control directives may be used to test status prior to job continuation.

File System

CODE	DESCRIPTION
FS01	Unrecoverable I/O error to the System Master Directory (SMD).
FS02	Unrecoverable I/O error to a disc allocation map.
FS03	Attempt to add a new file, but the System Master Directory (SMD) 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	Reserved.
FS08	Unrecoverable I/O error occurred while zeroing a file during creation.

Fortran

CODE	DESCRIPTION
FT01	Fortran scratch file *U1 must be expanded (i.e., $ASSIGN3 *U1=DC,????).
FT02	Fortran scratch file *U2 must be expanded (i.e., $ASSIGN3 *U2=DC,????).
FT03	Binary output (BO) file must be expanded if a disc file (i.e., $ASSIGN2 BO=SLO,???? or a direct assignment to the card punch - $ASSIGN3 BO=CP).
FT04	The compiled program caused the SGO file to overflow. The size of the SGO file can be increased via SYSGEN or may be assigned to tape via Operator Communications.
FT05	End of medium on nonspooled SLO file. File must be expanded if a disc file or a direct assignment made to the line printer - $ASSIGN3 LO=LP.
FT06	Potential abort conditions have been detected during program compilation. An abort status flag within the job's TSA has been set during compiler termination processing. Conditional job control directives may be used to test status prior to job continuation.

CODE	DESCRIPTION
HT01	Trap occurred in user's map area.
HT02	Trap occurred in another interrupt trap routine.
HT03	Trap occurred while no other tasks were in the active state.
HT04	Reserved.
HT05	User was unmapped when trap occurred.

CODE	DESCRIPTION
IO01	An I/O operation has been attempted for which the FCB is not properly linked to a File Assignment Table (FAT) entry. Since this linkage is established by IOCS when the file is opened, either the user task has not properly opened the file or the FCB has been inadvertently destroyed subsequent to the time the open file operation was performed.
IO02	An I/O operation has been attempted on an unopened file. This abort code will normally be issued when a user has opened a file, subsequently closed the file, then attempted an I/O operation on the file.
IO03	An unprivileged task is attempting to read data into an area of core which is not allocated for its use. This type of abort is usually caused by an invalid TCW in the task's FCB.
IO04	The control specifications in the FCB specify random access. However, the random access address contained in the FCB does not fall within the limits of the file.
IO06	Invalid blocking buffer control cells have been encountered during a read operation performed on a blocked file. This type error is normally caused in one of the three ways: (1) The user's blocking buffer has been inadvertently destroyed. (2) The file being read is not a properly blocked file. (3) A data transfer error has occurred on input of data from the file.
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	Reserved
IO09	The task has attempted to perform a rewind operation on the system SYC file.
IO10	The task has attempted a write End-of-File operation on a file which has been opened in the Read-Only access mode.
IO11	The task has attempted a write End-of-File operation on the system SYC file.

IO12	The task has attempted an erase or punch trailer operation on a file which has been opened in the Read-Only access mode.
IO13	The task has requested an illegal operation to be performed on a system file (backspace file, upspace, erase or punch trailer, eject, advance record, advance file, or backspace record).
IO14	A task running in the unprivileged mode has attempted to reserve an I/O channel.
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.
IO17	The task has attempted an open operation on a file, and no File Pointer Table (FPT) entry exists with a matching file code. This type of abort is most often caused by an improper or missing file assignment directive at catalog or linking load time. This type abort may also occur if the logical file code portion of the task's FCB has been inadvertently destroyed.
IO18	Reserved
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.
IO24	A task has specified an illegal address or transfer count in the FCB TCW. This type of error is usually the result of trying to output to a halfword device from a data area which is not on a halfword boundary. This error may also occur if the task attempts to transfer other than an even multiple of halfwords to or from a halfword device.
IO25	The task has requested a data transfer operation (read or write) with a Transfer Control Word (TCW) which specifies a quantity of zero.
IO26	Illegal sequence of operations while in read mode on either a system file or a blocked file.
IO27	Illegal sequence of operations while in write mode on either a system file or a blocked file.
IO28	Attempt to advance a record while in the write mode on a blocked file.
IO29	Attempt to advance a file while in the write mode on a blocked file.
IO30	Illegal or unexpectecd volume number encountered on magnetic tape.

IO32	Calling task has attempted to perform a second read on a $ statement through the SYC file.
IO33	An Invalid Device Address has been specified in the Task's Input/Output Control Header (IOCH).
IO34	An unprivileged task has requested the link service.
IO35	An unprivileged task has specified an IOCB list greater than 30 IOCB's in length.
IO36	A SYSGEN error has occurred, and the handler HAT address is not in the Controller Definition Table (CDT).
IO37	Job sequence number not found in the job table for task attempting to open SYC or SGO file.
IO38	The task has requested a write operation to be performed on a file which has been opened in the read-only access mode. Permanent files to which a task has read but not write access are opened read-only even though read-write is specified when the file is opened.
IO39	Blocked file indicated in FCB (or implied via assignment to a system file) but no blocking buffers available.
IO40	User TCW is in error due to one or more of the following conditions: 1. Unable to construct a valid TCW because the transfer count is too large. 2. Transfer count not an even multiple of transfer type. 3. Data address not bounded for transfer type (types = W, HW, B).
IO43	Input/Output Control List (IOCL) or data address not in contiguous 'E' memory (ASYNC,BSYNC).
IO46	Dynamic storage space for IOCDs within IOQ exhausted.
IO47	Class 'E' device TCW is not in class 'E' memory. This type of error indicates a map failure.
IO48	Reserved
IO49	Device access failure on OPEN.
IO53	The user has attempted to write to SYC file in Batch mode.
IO54	An attempt has been made to use the same logical file code in two or more File Control Blocks.

CODE	DESCRIPTION
JC01*	Unrecoverable read error from job's SYC file.
JC02*	Unrecoverable write error on SLO file.
JC03*	Unrecoverable write error on job's SGO file.
JC04*	Unable to build FAT/FPT for SLO or for SBO link file which indicates a program error.
JC05*	Unable to allocate disc space for SLO file.
JC06*	An entry is not available in the System Output Directory (M.SOD) for the definition of the job's SLO or SBO link file.
JC07	This job's Job Table entry has been destroyed which indicates a program error.
JC08*	Unable to allocate the job's SYC file.
JC09	Unrecoverable I/O error to SMD returned on call to File System Executive (H.FISE,10).
JC10*	Unrecoverable I/O error to disc allocation map returned by File System Executive (H.FISE,3 or H.FISE,4).
JC11*	Unable to allocate job's SGO file.

* Whenever a Job Control task aborts with one of these codes, the associated job is deleted.

CODE	DESCRIPTION
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.

CODE	DESCRIPTION
LE01	Directive error. The last directive printed on the directive list is in error.
LE02	Object record sequence error. The name of the module with the error will be the last line printed on the log.
LE03	Object record checksum error. The name of the module with the error will be printed as the last line of output on the log.
LE04	Object module format error. The module whose name is the module following that one has an invalid record code in record 1.
LE05	Incomplete object module. An object module, whose name is the last printed line of the log, has no terminal record (Hex DF).
LE06	Unrecoverable error on the SYC file.
LE07	Unrecoverable error on the LGO file.
LE08	Unrecoverable error on the LLO file.
LE09	Unrecoverable error on the LIB file.
LE10	Unrecoverable error on the DIR file.
LE11	Unrecoverable error on either the dynamically assigned temporary library file or temporary directory file.
LE12	Allocation denial on request for temporary disc space used to perform update function (i.e., disc space unavailable).
LE13	Delete table overflow. A combined maximum of 255 modules may be deleted/replaced/added in any one LIBED run.
LE14	End-of-medium encountered on temporary library. This error indicates the need to expand the subroutine library assigned to the LIB file code. The FILEMGR may be used to perform this function prior to another LIBED run.
LE15	End-of-medium encountered on temporary directory. This error indicates the need to expand the subroutine library directory assigned to the DIR file code. The FILEMGR may be used to perform this function prior to another LIBED run.
LE16	End-of-medium encountered on LIB file. The assigned file must be reallocated.
LE17	End-of-medium encountered on DIR file. The assigned file must be reallocated.

LE18	End-of-medium encountered on either LIB or DIR file. This error may occur on either a log, statistics, or update run and indicates that a previous CREATE run terminated prior to completion with an uncorrectable I/O error or a LE16/LE17 error.

MEDIA

CODE	DESCRIPTION
MD01	Potential abort conditions have been detected during media conversion operation. An abort status flag within the job's TSA has been set during compilation or execution processing. Conditional job control directives may be used to test status prior to job continuation. See output on logical file Code *OT for details about the abort condition.
MD02	At EOF on a SLO file.

MACLIBR

CODE	DESCRIPTION
ME99	Potential abort conditions have been detected during library editing operation. An abort status flag within the job's TSA has been set during editing processing. Conditional job control directives may be used to test status prior to job continuation.

Memory Parity Trap (H.IP02)

CODE	DESCRIPTION
MP01	Memory error occurred in a task's logical address space.
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.

CODE	DESCRIPTION
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	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).
MS03	A privileged task bit Set/Reset address is outside of the operating system or a static memory partition, or an unprivileged task bit Set/Reset address is outside of a static memory partition.
MS04	Task has attempted to create a timer entry to request an interrupt with a priority level outside the range of X'12' to X'7F', inclusive, or the requesting task is unprivileged.
MS05	Invalid function code has been specified for request to set user status word.
MS06	Unprivileged task has attempted to reset a task priority level or a privileged task has attempted to reset a task priority to a level outside the range of 1 to 64, inclusively.
MS07	Cannot load overlay segment due to software checksum or data error.
MS08	Overlay is not in the SMD.
MS09	Task has attempted to connect a task to an interrupt level not defined for indirectly connected tasks.
MS10	Overlay has an invalid preamble.
MS11	An unrecoverable I/O error has occurred during overlay loading.
MS12	Overlay is password protected.
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.
MS21	Multi-volume 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 (0 if tape is multivolume; non-zero if tape is single volume).
MS25	Operator has aborted task in response to mount message.
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	Task has attempted to load the interactive Task Debugger overlay in a memory-only environment.
MS30	Task has attempted to obtain a permanent file log in a memory-only environment.
MS31	User attempted to go to an any wait state from an end action routine.
MS32	Invalid register set-up detected in M.ID.
MS89	An unprivileged task has attempted to reestablish an abort receiver (other than M.IOEX).
MS90	Task has made a run request end action routine exit while the run request interrupt was not active.
MS91	Task has attempted normal exit with a task interrupt still active.
MS92	Task has attempted to queue a message during its exit sequence.
MS93	An invalid Receiver Exit Block (RXB) address was encountered during message exit.
MS94	An invalid Receiver Exit Block (RXB) return buffer address was encountered during message exit.
MS95	Task has made a message exit while the message interrupt was not active.
MS96	An invalid Receiver Exit Block (RXB) address was encountered during run receiver exit.
MS97	An invalid Receiver Exit Block (RXB) return buffer address was encountered during run receiver exit.
MS98	Task has made a run receiver exit while the run receiver interrupt was not active.
MS99	Task has made a message end action routine exit while the message interrupt was not active.

CODE	DESCRIPTION
RS47	Invalid time interval request.
RS48	Invalid activation request.
RS49	Invalid run request.
RS53	Invalid task number.
RS60	Invalid address specified.
RS65	Invalid delete request.
RS66	Invalid abort request.
RS67	Invalid resource mark request.
RS68	Invalid disconnect request.
RS69	Skip file or record operation requested on non-existent FCB.
RS70	Allocation error (appears only if IOSTAT and $n parameters have been omitted).
RT01	Unformatted read I/O error.
RT02	Formatted read I/O error.
RT03	Unformatted write I/O error.
RT04	Formatted write I/O error.
RT05	Reference made to non-existent device type or address.
RT06	Unit out of 0-999 range.
RT07	No left parenthesis on format.
RT08	Transfer index out of range (option 7 or M:ERRFLG can be used to avoid an abort).
RT09	Format error.
RT10	The I/O transfer requirements for the data buffer are incompatible with the amount of available data.
RT11	Format parenthesis level in excess of two.
RT13	Argument list exceeds logical read record.

RT14	Incorrect descriptor in format.
RT15	Integer descriptor but non-integer argument (option 7 or M:ERRFLG can be used to avoid an abort).
RT16	Hexadecimal descriptor but non-hexadecimal argument (option 7 or M:ERRFLG can be used to avoid an abort).
RT17	D, E, F, G descriptor, not real or complex argument (option 7 or M:ERRFLG can be used to avoid an abort).
RT18	Logical descriptor but non-logical argument (option 7 or M:ERRFLG can be used to avoid an abort).
RT19	Attempt to read past EOF/EOM.
RT20	Attempt to write past EOF/EOM.
RT21	Attempt to read past EOF/EOM.
RT22	Attempt to write past EOF/EOM.
RT23	Attempt to backspace following EOF/EOM.
RT24	Rewind after EOF/EOM.
RT25	Formatted record read.
RT26	Unformatted record read.
RT27	Doubleword integer overflow (option 7 or M:ERRFLG can be used to avoid an abort).
RT28	Byte integer input with negative sign (option 7 or M:ERRFLG can be used to avoid an abort).
RT29	Byte integer overflow (option 7 or M:ERRFLG can be used to avoid an abort).
RT30	Halfword integer overflow (option 7 or M:ERRFLG can be used to avoid an abort).
RT31	Full word integer overflow (option 7 or M:ERRFLG can be used to avoid an abort).
RT32	Illegal character in D, E, F, G input (option 7 or M:ERRFLG can be used to avoid an abort).
RT33	Underflow in floating conversion (option 7 or M:ERRFLG can be used to avoid an abort).
RT34	Overflow in floating conversion (option 7 or M:ERRFLG can be used to avoid an abort).

RT35	Argument list overflow (option 7 or M:ERRFLG can be used to avoid an abort).
RT36	Argument list overflow (option 7 or M:ERRFLG can be used to avoid an abort).
RT40	Attempt to free busy IOCH/IOCB entry.
RT41	Attempt to link busy IOCH/IOCB entry.
RT42	IOCH/IOCB table overflow.
RT43	Wait I/O returned before I/O termination.
RT44	Status parameter not linked to ADI device prior to I/O request.
RT46	ADI table address not on halfword boundary.
RT50	Missing or omitted parameter.
RT51	Parameter out of range.
RT52	End of search list reached.
RT55	Error found in math library routine.
RT61	List-directed I/O (input) encountered, character string split between two records.
RT62	Internal file read/write past EOF/EOM with no END option specified.
RT63	Block number exceeds maximum block number in file.
RT64	Record overflow.
RT65	Record length exceeds maximum allowable.
RT66	Record length not specified for random access or specified for sequential file.
RT67	Implicit open not allowed for or random access I/O.
RT68	Reference to sequential operation on a file opened for direct access not allowed.
RT69	Error(s) encountered on open.
RT80	Subscript error (i.e., subscript not a decimal number, illegal punctuation, excessive subscripts, or subscript out of range).
RT81	NAMELIST identifier error (i.e., column 1 non-blank, ampersand character not present, name does not immediately follow ampersand character, or non-blank following name).

RT82	Symbolic name error (no equal sign after variable/array name).
RT83	Data item error (i.e., excessive values for symbol or expected to find symbol).
RT84	Illegal value (i.e., illegal punctuation, missing comma, zero Hollerith count, or illegal character in value).
RT85	Attempt to read past EOF/EOM.
RT86	Attempt to write past EOF/EOM.
RT87	Symbolic name not defined in NAMELIST statement.
RT88	Repeat count error.
RT89	Symbolic name exceeds eight characters.
RT90	Invalid read/write operation.
RT91	End-of-file status return pursuant to random access record.
RT92	Random access partition number out-of-range (i.e., partition number not between 1 and 95, inclusive).
RT93	Random access number out-of-range (i.e., record number not between 1 and 65,535, inclusive).
RT94	Random access transfer length (write/read) or record size definition (define) out-of-range (i.e., transfer record length not between 1 and 65,535 bytes, inclusive).
RT95	Invalid random access argument list length.
RT96	FCB table overflow (16 or more files for RTM; 31 or more files for MPX-32).
RT97	Diagnostic output message exceeds 100 lines. To allow more diagnostic messages, statically assign the DO file (i.e., $ASSIGN2 DO=SLO,500).
RT98	Denial return when attempting to allocate file for diagnostic output message.
RT99	Insufficient blocking buffer space (each unit assignment to a system file requires one blocking buffer unless one file is assigned to another, i.e., via $ASSIGN4).

System Binary Output

CODE	DESCRIPTION
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.

System Check Trap Processor

CODE	DEFINITION
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).

CODE	DESCRIPTION
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.
SG14	Unable to allocate disc space for SYSTEM IMAGE file.
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 80K 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.
SG23	Insufficient shared memory table entries specified with SHARE directive. Number of entries must be equal to or greater than the number of partitions specified with /PARTITION NAME directives.
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 SMD, SWP, SID, LOD or POD directives is not included in the configuration being built.
SG28	The null device specification which is required to be included in every configuration is missing.
SG99	Directive errors encountered.

CODE	DESCRIPTION
SM01	The System Input Directory (M.SID) which is created at SYSGEN, does not exist. The directory may be created with the File Manager, but the file must be zeroed during creation.
SM02	The Systems Output Directory (M.SOD) which is created at SYSGEN, does not exist. The directory may be created with the File Manager, but the file must be zeroed during creation.
SM03	Unable to build a FAT/FPT for a system output task which is attempting to allocate an SLO or SBO file. Indicates a program error.
SM04	The Job Table entry associated with a job for which end-of-job processing is being performed has been destroyed. Indicates a program error.
SM05	Entry linkage is not consistent on the System Output Directory (M.SOD). The contents of M.SOD have been destroyed or a program error exists.
SM06	Entry linkage has been destroyed on the System Input Directory (M.SID).
SM07	Entry linkage has been destroyed on the System Output Directory (M.SOD).
SM08	Unrecoverable I/O error on spooled link file.
SM09	Unrecoverable I/O error on System Input Directory (M.SID).
SM10	Unrecoverable I/O error on System Output Directory (M.SOD).
SM11	Unrecoverable I/O error to a disc allocation map returned on call to File System Executive (H.FISE,4).
SM12	Attempt to activate System Output task unsuccessful.
SM13	Unrecoverable I/O error to the SMD returned on call to File System Executive (H.FISE,1).
SM14	Attempt to access a system input or output file in a memory-only environment.

System Input Task (J.SSIN)

CODE	DESCRIPTION
SN01	Blocking buffer or FAT space is not available.
SN02	Unrecoverable I/O error from the disc file being used as the SYC file.
SN03	System Input Directory (M.SID) does not exist or an unrecoverable I/O error was encountered in attempting to access it.
SN04	Job Sequence Number has been duplicated. Indicates a program error.
SN05	Spooled Input Directory (M.SID) is full.
SN06	A permanent file specified on the OPCOM BATCH command does not exist.
SN07	Unrecoverable I/O error to the SMD returned on call to the File System Executive (H.FISE,1 or H.FISE,10).
SN08	Unrecoverable I/O error to the allocation map returned on call to the File System Executive (H.FISE.3 or H.FISE,4).

System Output Task (J.SOUT)

CODE	DESCRIPTION
ST01	Unrecoverable write error on destination device for SLO or SBO records.
ST02	Unable to perform file code to file code allocation for 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.

SVC Trap Processor (H.IP06)

CODE	DESCRIPTION
SV01	Abort of unprivileged task using M.CALL.
SV02	Invalid SVC number abort.
SV03	Abort of unprivileged task attempting use of a "privileged-only" service.
SV04	Invalid SVC type abort.
SV05	Abort of unprivileged task attempting M.RTRN.

Swap Scheduler Task (J.SWAPR)

CODE	DESCRIPTION
SW01	Unrecoverable I/O error.
SW02	Reserved
SW03	Reserved
SW04	No 'E' memory available for SWAPR's buffer file.
SW05	No FAT or FPT to allocate.
SW06	Task has requested inswap but was never outswapped.
SW07	EOM detected on swap file.

'SYSBUILD'

CODE	DESCRIPTION
SY01	Unable to allocate or open input device during initial task loading. (Memory only MPX-32)
SY02	Unable to activate task. (Memory only MPX-32)
SY03	Unable to deallocate or close input device after initial task loading.
SY04	IPL device is undefined.
SY05	File is too small for the tape contents.
SY06	Transfer count on read is zero.
SY07	Unable to create a permanent file.
SY08	Unable to allocate file.

CODE	DESCRIPTION
UD01	Potential abort conditions have been detected during update processing. An abort status flag within the job's TSA has been set during execution processing. Conditional job control directives may be used to test status prior to job continuation.
UD02	User requested abort from mount prompt.

Miscellaneous Abort Codes

CODE	DESCRIPTIONS
BT01	Block mode timeout trap.
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 request.
MC01	Machine check trap.
MF01	A map fault trap has occurred. This is the result of a bad memory reference outside of the user's addressable space.
MP01	Memory error occurred in a task's logical address space. This is an internal or CPU failure. Rerun task.
NM01	Indicates a CPU failure.
OC01	The operator has requested that the task be aborted.
PV01	Privilege violation trap.
TS01	User requested removal from a BREAK request.
TS02	User requested removal from a Wait State queue.
TS03	Task running from specified terminal was aborted when the terminal disconnected.
UI01	Undefined instruction trap.

When system crash occurs as a result of a trap handler entry, the CPU halts with the registers containing the following information:

R0=PSD Word 0 (when trap generated)
R1=PSD Word 1 (when trap generated)
R2=Real address of instruction causing trap
R3=Instruction causing trap
R4=CPU status word (from trap handler)
R5=Crash code:

MP01=X'4D503031'	(See H.IP02 Codes)
NM01=X'4E4D3031'	(Non-Present 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'42543031'	(Cache Parity - H.IP10) 32/87 only
BT01=X'42543031'	(Block Mode Timeout - H.IP0E)
HT01=X'48543031	(Privileged Halt Trap - H.IPHT) CONCEPT/32 only
SW01=X'53573031'	(See SWAPR codes)

R6=Real address of register save block
R7=C'TRAP'=X'54524150'

For further description, see Volume 1, Section 2.10.

APPENDIX D

NUMERICAL INFORMATION

TABLE OF POWERS OF TWO

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
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	0.000 000 000 000 000 000 108 420 217 248 550 443 400 745 280 086 994 171 142 578 125

APPENDIX E

POWERS OF INTEGERS

TABLE OF POWERS OF SIXTEEN

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^{-12}
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}

TABLE OF POWERS OF TEN

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 AA32 36A4 B449 x 16^{-14}
DF0 B683 A764 0000	18	0.1272 5DD1 D243 ABA1 x 16^{-14}
8AC7 2304 89E8 0000	19	0.1D83 C94F B6D2 AC35 x 16^{-15}

APPENDIX F

ASCII INTERCHANGE CODE SET

Row	Col	0	1	2	3	4	5	6	7
Bit Positions 4 5 6 7	0 1 2 3	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	1 1 0 1	1 1 1 0	1 1 1 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	DCI 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	] 11-8-2	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

Some positions in the ASCII code chart may have a different graphic representation on various devices as:

ASCII	IBM 029
!	!
[	¢
]	!
^	>

Control Characters:

NUL	–	Null	DC3	–	Device Control 3
SOH	–	Start of Heading (CC)	DC4	–	Device Control 4 (stop)
STX	–	Start of Text (CC)	NAK	–	Negative Acknowledge (CC)
ETX	–	End of Text (CC)	SYN	–	Synchronous Idle (CC)
EOT	–	End of Transmission (CC)	ETB	–	End of Transmission Block (CC)
ENQ	–	Enquiry (CC)	CAN	–	Cancel
ACK	–	Acknowledge (CC)	EM	–	End of Medium
BEL	–	Bell (audible or attention signal)	SS	–	Start of Special Sequence
BS	–	Backspace (FE)	ESC	–	Escape
HT	–	Horizontal Tabulation (punch card skip) (FE)	FS	–	File Separator (IS)
LF	–	Line Feed (FE)	GS	–	Group Separator (IS)
VT	–	Vertical Tabulation (FE)	RS	–	Record Separator (IS)
FF	–	Form Feed (FE)	US	–	Unit Separator (IS)
CR	–	Carriage Return (FE)	DEL	–	Delete
SO	–	Shift Out	SP	–	Space (normally nonprinting)
SI	–	Shift In	(CC)	–	Communication Control
DLE	–	Data Link Escape (CC)	(FE)	–	Format Effector
DC1	–	Device Control 1	(IS)	–	Information Separator
DC2	–	Device Control 2			

APPENDIX G

IOP PANEL COMMANDS

AS (CR)	Clear address stop
AS=xxxxxxxx (CR)	Set address stop
CRMD =xxxxxxxxxxxx (CR)	Load CRAM with xxxxxxxxxxxx
=xxxxxxxxxxxx (CR)	Load CRAM with data and increment address
CS (CR)	Read control switches
CS=xxxxxxxx (CR)	Set control switches
EA (CR)	Read effective address
EXEC (CR)	Execute CRAM
GPR (CR)	Read general purpose registers
HALT (CR)	Halt
IPL (CR)	IPL from default address
IPL=xxxx (CR)	IPL from xxxx
IS (CR)	Clear instruction stop
IS=xxxxxxxx (CR)	Set instruction stop
MA=xxxxxx (CR)	Read physical memory address location
(CR)	Increment and read memory address
MAV=xxxxxx (CR)	Read virtual memory address location
(CR)	Increment and read memory address
MD=xxxxxxxx (CR)	Write memory data
=xxxxxxxx (CR)	Increment and write memory data
= (CR)	Increment and write previous data
MSGE (CR)	Message
OVR (CR)	Toggle clock override
PRIP (CR)	Set primary panel (master only)
PSD (CR)	Read Program Status Doubleword (1 and 2)
PSD=xxxxxxxx (CR)	Write Program Status Word (2)
PSW (CR)	Read Program Status Word (1)
PSW=xxxxxxxx (CR)	Write Program Status Word (1)
REGA=xxxxxxxx (CR)	Write General Purpose Register A
RS (CR)	Clear read operand stop
RS=xxxxxxxx (CR)	Set read operand stop
RST (CR)	Reset
RUN (CR)	Run
SECP (CR)	Set secondary panel (master and slave)
STEP (CR)	Instruction step
(CR)	Instruction step
WS (CR)	Clear write operand stop

WS=xxxxxxxx (CR)	Set write operand stop
@@C	Enter console mode
@@P	Enter panel mode
(LF)	Repeat command

Notes:

1 (CR) denotes Carriage Return after each command.

2 LOCK ON and LOCK OFF are not supported by the CRT panel.

Console Mode

To change from SCP mode to console mode, it is necessary for an operator to input @@C (CR).

Upon receipt of the CR command 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 control panel mode, enter @@P (CR). When the control panel mode is selected, // is used as the prompt.

Glossary

Address Space	The range of addresses described by a given configuration of the SELMAP. Normally the address range is from 0 to some map block boundary. Also referred to as "logical address space". When the CPU is unmapped, the only address space is the range of physical memory configured.
Batch	Batch processing is the sequential processing of multiple jobs, with the operating system's Job Control program providing automatic job-to-job transition. Job Control and batch are often used synonymously.
Indirectly-Connected Interrupt	A hardware interrupt for which a special control block is defined at SYSGEN. The control block is connected to an active software task via the M.CONN service or OPCOM CONNECT command so that when the hardware interrupt occurs, the task is resumed. Resumption is based on the software priority of the user task.
Console	The teletypewriter or CRT terminal which is used by the operator to control the operation of MPX-32. Also called "operator's console", "OPCOM console", or "system console".
Context Area	Task program Status Doubleword (PSD) and registers stored in the Task Service Area.
Context Switch	The process of one task relinquishing CPU control and another task gaining CPU control. Software: based on availability of resources and software priorities. Hardware: based on hardware interrupt and trap priorities.

Controller Definition Table (CDT)	The CDT contains an entry for each controller described to SYSGEN. This entry describes the channel number, device type code, the number of units on the controller, Unit Definition Table (UDT) address, etc. IOCS uses the CDT entry to record controller status and control I/O queue information.
DATAPOOL	Datapool is a memory partition which can be defined at SYSGEN or via the FILEMGR. DATAPOOL structure is defined by a datapool dictionary created and maintained via the Datapool Editor utility (DPEDIT). Elements of DATAPOOL storage are referenced symbolically rather than by their address.
DATAPOOL Dictionary	The DATAPOOL dictionary is a map of elements in the DATAPOOL. The dictionary equates a logical location in the DATAPOOL to a symbolic value.
DATAPOOL Editor	The DATAPOOL Editor (DPEDIT) is a utility used to maintain the DATAPOOL dictionary. Definitions of elements in the DATAPOOL can be modified, deleted, or added to the dictionary via the DATAPOOL Editor.
Device Handlers	Device handlers execute queued I/O commands, process service interrupts, and perform device testing functions. A handler is provided for each standard peripheral device.
Directly-Connected Interrupt	A user module that is incorporated with the resident MPX-32 operating system at SYSGEN and associated with a specific hardware interrupt level; hardware context switch on interrupt goes directly to this task without routing through MPX-32.
Disc Allocation Unit	A disc allocation unit is a number of records, the size of which is disc-dependent. There are currently between one and four 192-word blocks per disc allocation unit.
Dispatch Queue	A list of tasks that have been activated, their priorities, and other pertinent information. Referenced by the Execution Scheduler for software context switches.
Fast File	The file definition is retrieved from the System Master Directory (SMD) with one disc access.

File	A collection of related records on a recording medium. MPX-32 input files may be on disc, tape, paper tape, or cards, or input can come from a terminal. Output files can be directed to the above as well as line printers. "File" in MPX-32 normally denotes disc files (as opposed to files on a device medium).
File Assignment Table (FAT)	A FAT describes the channel and the device to which a file is assigned. There is a FAT entry for each logical file code referenced by a user program.
File Code	A three-character field in a File Control Block (FCB) that contains a mnemonic that refers to the file. The mnemonic can be used to refer to the FCB in Job Control statements, TSM and OPCOM commands, and system services such as $ASSIGNx. Synonymous with logical file code.
File Control Block (FCB)	An FCB contains parameters which describe an I/O operation and identify the file code on which the operation is to be performed. User tasks and system programs define the contents of their related FCB's.
File Manager	The File Manager (FILEMGR) is a utility program used to maintain permanent disc files. It provides a set of directives for creating, deleting, saving, and restoring files.
File Name	A file name is the one- to eight-ASCII character name of a permanent file. It is established when the file is created.
File Pointer Table (FPT)	The FPT provides the link between a user's FCB identifying the file and the FAT entry describing the device to which the user's file code is assigned. Static contents of the FPT are built by the MPX-32 resource allocation service; they are updated dynamically by the device allocation and deallocation system services. The FPT table is maintained in the task's TSA built by MPX-32.
File System Executive (FISE)	The File System Executive provides the interface between user tasks and available file space. It performs dynamic allocation and deallocation of temporary file space, locates or adds permanent file definitions in the System Master Directory, and maintains the disc allocation map.

Global Common	Global common is a set of memory partitions defined via SYSGEN or the FILEMGR which multiple tasks can access as a global resource. Definitions of Global Common items are determined by their relative position within the common partition.
I/O Control System	The Input/Output Control System (IOCS) processes I/O requests by the task and queues these requests by the software priority of the task.
Impure Data	Tables, arrays, etc. that are an integral part of a task, yet can be modified during execution. If a task is sectioned, impure data are located in DSECT and reside in read/write memory separate from code and pure data, which are located in CSECT and reside in read-only memory.
Independent	A task that runs independent of an interactive or batch environment, normally a real-time task.
Interactive	Of or pertaining to terminals. An interactive task is one doing I/O to a terminal.
Interrupt	An internal or external event that requires rapid service by special software (or firmware) routines. The CPU preserves its current state and transfers control to the required interrupt handler.
Interrupt Control Block (ICB)	A series of memory locations included in a software interrupt handler that are pointed to by the IVL. This is where the CPU preserves its current state and transfers control to a software handler.
Interrupt Vector Location (IVL)	Dedicated memory location for connecting an interrupt to connecting an ICB.
Job	A job is a sequential set of tasks whose batched execution is scheduled by the Job Control program based on Job Control statements.

Job Control Program	A program (i.e., a collection of tasks) which provides batch processing services, including: job-to-job transition, management of spooled I/O, run-time I/O device assignments. Job Control processing is controlled through Job Control Statements. One copy of the Job Control program is used for each job processed concurrently in the system. The number of Job Control programs that can run concurrently is specified at SYSGEN time.
Job Name	A job name is the one- to eight-character job name from the $JOB statement which defines the beginning of the job. An owner name is also specified for the job.
Job Queue	The single ordered set of all SYC files containing spooled jobs waiting to be initiated by Job Control programs.
Job Sequence Number	A job sequence number is a decimal number which is assigned sequentially to a job when it is spooled to SYC disc space by the System Input program.
Job Stream	The sequence of jobs processed by a particular Job Control program. The number of job streams processed by MPX-32 is the number of Job Control programs which may run concurrently, which is established by SYSGEN. Synonymous with batchstream.
Key	A one-to-eight character code associated with an ownername/username to provide a degree of protection for system access and user files.
Load Module	A cataloged task that may be activated in one of three task activation environments: independent, interactive, or batch.
Load Module Information Table	A part of a load module produced by the Cataloger that contains special indicators for the task, a relocation matrix, and resource requirements.
Logical File Code (lfc)	Synonymous with file code (q.v.).
Macro Library Editor	The Macro Library Editor (MACLIBR) is a utility program used to create disc and tape resident macro libraries for use with the Macro Assembler.

Map	Synonymous with SELMAP (q.v.).
Map Block	The set of memory locations whose addresses are calculated from a single map register. All map blocks are the same size and begin on addresses which are multiples of their size. Map block size is determined by hardware map granularity. On the 32/7x, a map block is 8KW and on the 32/27, a map block is 2KW. In general, a map block is the smallest mappable quantum of physical memory.
Media Conversion	The Media Conversion utility (MEDIA) is used for media to media conversion, media copying, and media verification. It provides functions ranging from file duplication to merging media inputs into single or multiple media outputs.
MPX-32 Executive	The MPX-32 Executive schedules CPU processing for all tasks running under software priority levels. MPX-32 stands for Mapped Programming Executive.
Object Module	An object module is the smallest unit of a task that is output on the SGO by the Assembler or Compiler and is identified by name. It consists of relocatable object code to be processed by the Cataloger.
On-Line or Online	Synomymous with INTERACTIVE (q.v.).
Operator Communications	Operator Communications (OPCOM) commands allow the user to exercise system control through a set of commands entered via the console or a terminal.
Owner	One who owns a task. See owner name.
Owner Name	A one to eight ASCII character, left justified, blank filled name, maintained in the M.KEY file, if it exists, and associated with task activation.
Priority	Relative eligibility. Hardware priority refers to the priority scheme of external interrupts and service interrupts. Software priority refers to the priority scheme used by MPX-32 to resolve conflicting requests for resources. Priority for a task activated independent of the

batch or interactive environment is the priority established for the task when it is cataloged. The priority for a task activated interactively (via a RUN command) is the terminal priority established at SYSGEN. The priority for a task activated as part of a batch job is 64 unless modified via the OPCOM command URGENT.

Privileged

Privilege is a state of processing that allows access to a set of otherwise unexecutable "privileged" instructions, and to a privileged collection of system services. The privileged/ unprivileged state of a task is indicated when it is cataloged.

Procedure

Synonymous with 'code'.

Processor

A SYSTEMS program that a user can run to perform a related set of operations such as file maintenance, media conversion, FORTRAN compilation, etc. Also used synonymously with utility. Language processors are compilers, assemblers, or interpreters. Command processors are Job Control, TSM, and OPCOM.

Program

A program is a part of a task, a task, or a set of related tasks.

Program Status Doubleword (PSD)

Hardware registers that define the state of the CPU at any given time.

Protection

Memory protection: Write by unprivileged programs is not allowed. File protection: Use of passwords etc. that allow the user who creates a file to protect it, if desired.

Protection Granule

The smallest unit of memory within a map block that can be individually write locked. On the 32/7x, a protection granule is 512 contiguous words beginning on a 512-word boundary.

Pseudonym

A one-to-eight character name which provides more information about a task's environment than the task number/ownername, etc. For example, TSM uses the pseudonym TSM*terminal-number to identify a task by the terminal it is running on and Job Control uses the pseudonym .Odevmnc to identify a spooling task's target device.

Pure Data	Tables, arrays, etc. that are an integral part of a task and do not change during execution. Located with code. If a task is sectioned, pure data are included with code in CSECT and reside in read-only memory.
Real-Time	Any user task which is not activated in an interactive or batch environment is usually considered a real-time task. Real-time tasks are typically designed to respond to external stimuli (such as external interrupts) or to be executed periodically, as with a timer.
Reel Identifier	A 1-4 character name output with a magnetic tape mount message and also written to a tape along with the volume number if using multivolume tape.
Reentrancy	The logical property of procedural code which allows it to be executed completely asynchronously by multiple concurrent tasks.
Resident	In MPX-32, user's code (e.g., an interrupt handler) that is incorporated with the MPX-32 operating system at SYSGEN and is mapped into every user's address space with the operating system.
Resident (Locked in Memory)	A task that is non-swappable. It remains in memory until it exits or aborts.
Resource Allocator	The part of the resident system whose primary function is to allocate memory and peripherals.
Resourcemark	A numeric value used cooperatively by tasks to synchronize access to a common resource such as a disc file or sharable device.
Resource Requirement Summary (RRS)	A part of the Load Module Information Table that defines the devices, files, etc. required for the task.
Scheduler	The CPU scheduler dispatches CPU control based upon system events and task priorities. Synonymous with MPX-32 Executive.

Slow File	On MPX-32, a slow file is one in which the scatter- storage mechanism used to build SMD entries may use a back-up algorithm if the file name maps into an existing active file (collision mapping). If the back-up algorithm is used, it may require additional disc access.
Source Update	The Source Update (UPDATE) utility is used to create and update user and system source files.
Subroutine Library Editor	The Subroutine Library Editor (LIBED) is a utility program for creating and updating subroutine libraries.
Supervisor Call (SVC)	Supervisor Call is a trap that provides user interface to system services. Also the name of the software instruction that causes the trap.
System General Object (SGO) File	SGO files are used for the accumulation of object code within batch jobs. A separate SGO file is allocated for each job and exists for the duration of the job.
System Binary Output (SBO) File	An SBO file is a temporary file used for punched output. SBO files generated by real-time tasks are output to destination peripheral devices when the files are deallocated. SBO files generated by a batch job are output to destination peripheral devices upon job completion.
System Control File (SYC)	An SYC file is a disc file that provides intermediate storage for Job Control statements, object code, and data for a batch job. A separate job file is dynamically created on the SYC for each task initiated in a user's job file, and when the last task in the job completes execution, the job's SYC is deleted.
System Generation (SYSGEN)	The System Generation program is used to tailor the MPX-32 operating system to the hardware and software requirements of an installation.
System Input	The System Input task transfers batch input from devices and dynamically linked temporary and permanent disc files to intermediate storage on System Control (SYC) files.

System Listed Output	An SLO file is a temporary file used for listed output. SLO files generated by real-time tasks are output to destination peripheral devices when the files are deallocated. SLO files generated by a batch job are output to destination peripheral devices upon job completion.
System Loader	The System Loader is the part of the Allocator that loads any cataloged load module into memory upon request; it performs all necessary biasing of relocatable data.
System Master Directory (SMD)	The SMD is created at SYSGEN and describes the location, length, name, and related information about each file or memory partition in the system. Entries are made in the SMD for load modules as a part of the cataloging process.
System Output	The System Output task outputs the print (SLO) and punch (SBO) data collected for each task from temporary disc files to destination listed and punched output devices.
System Service	Equivalent to the term "Monitor Service" in RTM.
Task	A task is a body of code which is scheduled for CPU time as a single entity. It has one Dispatch Queue Entry (DQE) and one Task Service Area (TSA). A task may be activated as real-time, batch, or on-line.
Task Identifier	Tasks are identified by attributes. Attributes of a task include its unique task number (see below), the name of its owner, its load module name, its job sequence number (if batch), and an optional pseudonym to use in intertask communication. The task number is the only attribute that can be used to abort or to communicate with a task that can be multicopied. It may be obtained by the user by specifying one or a combination of task attributes.
Task Number	A task number is a sequential 32-bit number which is assigned to the task when it is activated and identifies the task uniquely over time (until system restart). Task attributes known by the user can be used to obtain the task number. The operator must refer to the task only by task number when attempting to, for example, abort.
Task Scheduler	See Scheduler.

Terminal	An I/O device featuring a keyboard for input and either a CRT or a printer for output. The asynchronous communication channels for terminals are managed by the MPX-32 Terminal Services Manager (TSM).
Terminal Services Manager (TSM)	A collection of tasks and service routines which control the MPX-32 interactive environment.
Timer	An optional task scheduling mechanism provided for each task by MPX-32. Timers are managed using the same real-time clock which is used to maintain the time-of-day.
Timer Unit	A timer unit is a number of real-time clock interrupts selected by the user at SYSGEN to represent a logical unit of time.
Transfer Control Word (TCW)	A TCW contains byte count and data address information used to define and control an I/O operation involving data transfer.
Trap Processors	Most Trap Processors are firmware and/or software routines that are entered when any exceptional condition trap occurs; they then perform the appropriate processing. The SVC trap is a special case used to transfer control from a task to the MPX-32 operating system.
Unit Definition Table (UDT)	The UDT describes each peripheral device in the system. The entry describes the device subaddress and channel number, the device type code, unit status, number of allocation units, the physical characteristics of the device, etc.
Unprivileged	An unprivileged task is one that does not execute "privileged" instructions. The privileged/unprivileged state of a task is established when it is cataloged.
User	See User Name.
User Name	One to eight ASCII character, left justified, blank filled name, maintained in the M.KEY file, if it exists, and associated with all user files in the System Master Directory (SMD). The absence of a user name indicates a system file.

INDEX

This index is for use with the MPX-32 Reference Manual, Volumes I, II and III. Its format is section number (volume number) where the information can be found.