1700
1700
1700
1700

MSOS Version 3

CONTROL DATA
CORPORATION

INSTALLATION HANDBOOK

REVISION RECORD

REVISION	DESCRIPTION
12-70	Original Printing
Publication No. 60282700	

Address comments concerning this manual to:

Control Data Corporation
Technical Publications Department
4201 North Lexington Avenue
Arden Hills, Minnesota
or use Comment Sheet in the back of this manual.

PREFACE

MSOS 3.0 Products

The following products may be used with MSOS 3.0:

1700 Macro Assembler 2.0

1700 COSY 1.0

1700 Mass Storage FORTRAN 2.0A

1700 Mass Storage FORTRAN 2.0B

1700 System Checkout

1700 System Configuration

Installation Handbook Format

The purpose of the 1700 MSOS 3.0 Installation Handbook is to provide the information necessary for field analysts and systems programmers to install the system. It is assumed that these analysts and programmers have at least one year of system programming experience as well as introductory 1700 instruction. The Installation Handbook is divided into three parts:

Part I covers installation information (requirements, procedures, and modifications) of each of the MSOS product set members. Each section describes a separate product.

Section	
1	MSOS 3.0
2	Macro Assembler 2.0
3	COSY 1.0
4	Mass Storage FORTRAN 2.0A
5	Mass Storage FORTRAN 2.0B
6	System Checkout 1.0
7	System Configurator 1.0

Any add-ons will follow the same format.

Part II describes the system modules which must be changed to modify the standard system by adding drivers or optional modules. The second section of part II describes the installation of optional drivers; the third section describes the installation of optional modules.

Part III is an accumulation of installation related information which may be helpful in installation. Its sections are:

Section	
1	Conventions
2	Unit assignments
3	Hardware FCO levels
4	Initializer control statements
5	Initializer Procedures
6	Installation messages
7	Card deck and tape structures and contents

Additional MSOS 3.0 Related Manuals

1700 OPERATING SYSTEM OPERATING GUIDE	60191400
1700 MSOS 3.0 REFERENCE MANUAL	60282600
1700 COMPUTER SYSTEM MACRO ASSEMBLER REFERENCE MANUAL	60176300A
1700 COMPUTER SYSTEM MASS STORAGE/FORTRAN REFERENCE MANUAL	60192200A
1700 COSY/MSOS REFERENCE MANUAL	60237100
1700 CONTROL DATA 1700 COMPUTER SYSTEM CODES	60163500
1700 SYSTEM CHECKOUT REFERENCE MANUAL	60281800
1700 SYSTEM CONFIGURATOR REFERENCE MANUAL	60282300A

CONTENTS

PART I

INSTALLATION

PART II

CUSTOMIZATION

1.1 RELEASE DESCRIPTION

The release description of the standard released MSOS 3.0 includes new MSOS features, corrections, limitations, and deficiencies.

1.1.1 NEW FEATURES

- Optimization of mass memory: All I/O drivers, except mass memory drivers and the teletype-writer\keyboard are optionally mass memory resident. Also, even though the job processor and LIBEDT retain their 2.1 capabilities, their use of allocatable core is modified so that the size of allocatable core can be reduced.
- Engineering File: The engineering file logs all hardware errors on peripheral devices.
- MSOS 3.0 is capable of handling multiple devices on the 1706 buffered data channel.
- All MSOS PSRs received before July 1, 1970 are included in the MSOS 3.0 release.
- Core requirements for ODEBUG are less in MSOS 3.0 than in MSOS 2.1 because of sector addressing.
- Initializer capabilities: The input media options are expanded under 3.0 to include:

  1726-405 card reader

  1728-430 card reader

  1729-2 card reader

  1731/1732 magnetic tape unit

  1777 paper tape reader

- By using System Configurator, MSOS 3.0 may be customized for a specific system. Section 7.1.1 lists the new System Configurator features available for use with MSOS 3.0.
- FORTRAN monitor interface package, re-entrant I/O package and arithmetic routines are now standard products.
- The System Maintenance Routine (SMR) is now a standard utility package.

1.1.2 CORRECTIONS

All PSRs received before July 1, 1970 are included in MSOS 3.0. The PSR numbers are:

302	444	528	635
342	445	529	636
373	449	530	638
377	454	531	640
390	465	532-538	643
395	467	540	644
398	471	544	648
401	489	545	649
405	492	546-553	651
406	495	555	652
409	496	558-570	654
416	497	572-594	655
425	517	596	657
428	518	597	660
433	521	599-612	667
434	525	615-617	670
435	526	621-623	677
436	527	625-633	681

The following RSMs are also included in the MSOS 3.0 release:

A163
1531
1738

1.1.3 KNOWN LIMITATIONS

The SCN command in ODEBUG does not reject illegal hexadecimal values but converts them to zero and continues.

Statement editing for errors within the system initializer is limited. Incorrect commands can cause initialization malfunctions which require restarting the process to alleviate the problem.

Incorrect formatting of output to the TTY will result if the output message buffering package is used with Standard Recovery.

An *P statement in LIBEDT punches a single all-ones frame on paper tape even when no valid input is received.

The engineering file requires that the DISKWD driver be used.

1.1.4 KNOWN DEFICIENCIES

A program load operation which generates an E3 diagnostic will also generate an E13 diagnostic.

Loader blocks that are too long and are input to the system initializer hang it.

When replacing a file on the program library with a larger file under *N processor, the operation is performed. But, when listing the library under *DL the file name is printed twice, once where code should appear and again at the end of the listing.

COSY control card END/ is missing on PROTEC COSY source.

1.2 REQUIREMENTS

To install the standard MSOS 3.0 system, the necessary release materials, hardware and software must be available. These requirements are listed below.

1.2.1 RELEASE MATERIALS

The user receives the MSOS 3.0 system either on cards, magnetic tape or paper tape. Following are the materials issued to the user with the particular system he chooses as well as optional materials available to the user on request. The card deck and tape structures for these materials are in part III, section 7.1.

Card Version

System initializer and installable binaries card deck

System definitions and skeletons card deck

Magnetic Tape Version

One magnetic tape containing the system initializer and installable binaries

One magnetic tape containing system definitions and skeletons

Paper Tape Version

One system initializer paper tape

Eight installation paper tapes

Eleven paper tapes containing system definitions and skeletons

Optional

One COSY source magnetic tape

Source on cards

Three list magnetic tapes

1.2.2 HARDWARE REQUIREMENTS

Minimum Configuration

The installation procedures in part I, section 1.3 pertain to the following standard minimum machine configurations:

CONTROL DATA® 1704 Computer (with 4096 words of memory) or, CONTROL DATA® 1774 Computer

CONTROL DATA® 1705 Interrupt/Data Channel

CONTROL DATA® 1708 Storage Increment (3 with 4096 words of memory)

CONTROL DATA® 1711 Teletypewriter or CONTROL DATA® 1712 Teletypewriter

CONTROL DATA® 1738 Disk Controller

CONTROL DATA® 853 Disk Driver or CONTROL DATA® 854 Disk Drive

CONTROL DATA® 1777 Paper Tape Station

Optional Peripherals

Program operation can be enhanced by adding other peripherals. As peripherals are added and the system is expanded, the size of core storage must be expanded to accomodate the new drivers. Optional peripherals are listed below along with the sections in part II describing their installation instructions. The memory required for each is listed in part I, section 1.2.3.

	Section
CONTROL DATA® 1706 Buffered Data Channel	2.1
CONTROL DATA® 1726 Card Reader Control	2.2
CONTROL DATA® 405 Card Reader	
CONTROL DATA® 1728-430 Card Reader/Punch	2.3
CONTROL DATA® 1729-2 Card Reader	2.4
CONTROL DATA® 1751 Drum	2.6
CONTROL DATA® 1740 Printer Controller	2.7
CONTROL DATA® 501 Line Printer	
CONTROL DATA® 1731 Magnetic Tape Controller	2.8 and 2.9
CONTROL DATA® 1732 Magnetic Tape Controller	2.10
CONTROL DATA® 601 Magnetic Tape Transport	
CONTROL DATA® 608 Magnetic Tape Transport	
CONTROL DATA® 609 Magnetic Tape Transport	
CONTROL DATA® 1713 Teletypewriter	2.13
CONTROL DATA® 1572 Programmable Sample Rate Unit or CONTROL DATA® 1573 Line Synchronized Timing Generator	2.14

1.2.3 MEMORY REQUIREMENTS

If optional peripherals are to be added (part II, section 2), the following information explains the memory needed to add each driver. All lengths are in decimal.

Released Monitor

Core resident with mass memory drivers	6576
Allocatable core	1623

Available Drivers

Card Equipment	Core Resident	Mass Memory Resident
1726-405 card reader, buffered	378	404
1726-405 card reader, unbuffered	351	371
1728-430 card reader/punch	867	891
1729-2 card reader	440	460
Disk or Drum Equipment		
1738-853/854 disk	445	
1751 drum	272	
Line Printer Equipment		
1740-501 line printer	511	526
Magnetic Tape Equipment		
1731-601 buffered	1296	1311
1731-601 unbuffered	1286	1208
1732-608/609 buffered	949	1008
1732-608/609 unbuffered	938	986
Paper Tape Equipment		
1777 paper tape reader/punch	427	503
Teletypewriter		
1711/1712/1713 teletypewriter	319	
1713 reader/punch teletypewriter	607 + buffer size	

1.3 INSTALLATION PROCEDURES

1.3.1 SUMMARY

The installation of the standard released version of MSOS 3.0 consists of the following summarized steps. For detailed instructions, refer to the specified section (in part I).

1.	Load system initializer	if using card version: section 1.3.2 if using magnetic tape version: section 1.3.3 if using paper tape version: section 1.3.4
	a. Load the bootstrap which is on either cards, paper tape, or magnetic tape	
	b. Read and execute checksum loader	
	c. Execute system initializer	
2.	Install MSOS 3.0 operating system	section 1.3.5
	a. Set MAXCOR	section 1.3.5, step 2
	b. Set SECTOR	section 1.3.5, step 3
	c. If necessary, delete or add drivers	section 1.3.5, steps 4 and 5
	d. If necessary, reassign input, output, and/or comment devices	section 1.3.5, step 6
3.	If installing on a 1700 SC, enter the autoload program	section 1.3.6

System initialization messages are listed in part III, section 6.1; system installation messages are in part III, sections 6.2, 6.3, and 6.4.

1.3.2 LOAD CARD VERSION OF SYSTEM INITIALIZER

Mount disk pack on the disk drive.

Enter the Card Loading Sequence

Enter the loading sequence into core memeory beginning at core memory location 200. This code can be loaded at any location or run anywhere above the last location into which the checksum loader will load, but the location 200 is preferable. This sequence of code will read one formatted record (the checksum loader) into the location specified by the A register (which will be 0000).

1. Press the master CLEAR on the console
2. Set all switches to neutral positions
3. Set SELECTIVE STOP switch
4. Set P register
5. Set push button register to 200
6. Set ENTER/SWEEP switch to ENTER
7. Set X register
8. Enter the code in this manner:

   a. Enter first (or next word) of code into push button register

   b. Momentarily set RUN/STEP switch to STEP

   c. Press the display CLEAR button

   d. Proceed with each word in the X register column using steps a through c until all code is entered (The location is in the P register)

P Register	X Register	Mnemonic Instructions		
0200	681E	PRELOD	STA*	STADD
0201	0844	READ	CLR	A
0202	681A		STA*	NO
0203	60FF		STA-	I
0204	E000		LDQ	=N$0421
0205	0421			
0206	C000		LDA	=N$0080
0207	0080			
0208	03FE		OUT	-1
0209	0DFE		INQ	-1
020A	02FE	NEXT	INP	-1
020B	0FC8		ALS	8
020C	6811		STA*	TEMP
020D	02FE		INP	-1
020E	B80F		EOR*	TEMP
020F	6C0F		STA*	(STADD)
0210	D80E		RAO*	STADD
0211	C0FF		LDA-	I
0212	D0FF		RAO-	I
0213	09D8		INA	-39
0214	0121		SAP	LOOP-*-1
0215	18F4		JMP*	NEXT
0216	D806	LOOP	RAO*	NO
0217	C805		LDA*	NO
0218	09FC		INA	-3
0219	0131		SAM	DONE-*-1
021A	18E6		JMP*	READ
021B	18FF	DONE	NUM	$18FF
021C	0000	NO	NUM	0
021D	0000	TEMP	NUM	0
021E	0000	STADD	NUM	0
			END	PRELOD

9. Set the P register
10. The display should show 218_{16} which means that 25_{10} commands have been entered
11. Release SELECTIVE STOP switch

Check Loading Sequence

1. Press master CLEAR switch on console
2. Set all switches to their neutral position
3. Set SELECTIVE STOP switch
4. Set P register
5. Enter into the push button register the first core location to be examined
6. Set X register
7. Set ENTER/SWEEP switch to SWEEP position
8. Momentarily set the RUN/STEP switch to the STEP position

   The data in the core location specified in step 5 appears on the push button register
9. To display the next sequential word of core memory in the push button register, briefly set the RUN/STEP switch to the STEP position
10. Release SELECTIVE STOP switch

Read Checksum Loader

1. Place the system initializer and installable binaries deck in the card reader; ready the device
2. Set all swtiches to neutral
3. Press master CLEAR switch on console
4. Set the P register
5. Set push button register to 0200
6. Set the RUN/STEP switch to RUN

   The checksum loader, the first portion of the deck is read; the card reader stops processing.

Execute Checksum Loader

Since the initializer is the second portion of the system initializer and installable binaries deck (following the checksum loader), it is already in the card reader.

1. Set the A register
2. Set the push button register to xxxx

   xxxx is the length of MAXCOR minus the initializer length 1681_{16}.
3. Set the SELECTIVE STOP switch
4. Set the RUN/STEP swtich to RUN to load the tape
5. When the card deck stops processing cards, set the Q register
6. The push button register should be 0000
7. If the push button register does not read 0000, a checksum error occurred
   a. Re-insert the initializer portion of the card deck
   b. Return to step 1

Execute System Initializer

1. Press master CLEAR on console
2. Release SELECTIVE STOP switch
3. Set P register
4. Set the push button register to xxxx which is the address of the system initializer

   xxxx is the length of MAXCOR minus the initializer length.
5. Momentarily set the RUN/STEP switch to RUN
6. SI appears on the teletypewriter to indicate that the system initializer can now load the operating system. Continue installing with step 2, part I, section 1.3.5, Installing the Operating System.

1.3.3 LOAD MAGNETIC TAPE VERSION OF SYSTEM INITIALIZER

Mount the disk pack on the disk drive.

Enter the Magnetic Tape Loading Sequence

The bootstrap program reads in the absolute initializer which resides on magnetic tape. Enter the loading sequence into core memory beginning at core memory location 200. This code can be loaded at any location or run anywhere above the last location into which the checksum loader will load, but the location 200 is preferable. This sequence of code will read one formatted record (the checksum loader) into the location specified by the A register (which will be 0000).

1. Press master CLEAR switch on console
2. Set all switches to neutral positions
3. Set SELECTIVE STOP switch
4. Set P register
5. Set push button register to 200
6. Set ENTER/SWEEP switch to ENTER
7. Set X register
8. Enter the code in this manner:
   a. Enter first (or next word) of code into push button register
   b. Momentarily set RUN/STEP switch to STEP
   c. Press display CLEAR button
   d. Proceed with each word in the X register column using steps a through c until all code is entered (The location is in the P register)

P Register	X Register	Mnemonic Instructions		
0200	6832	MTLDR	STA*	HOLD
0201	E000		LDQ	=N$382
0202	0382			
0203	C000		LDA	=N$414
0204	0414			
0205	03FE		OUT	-1
0206	0DFE		INQ	-1
0207	C000		LDA	=N$100
0208	0100			
0209	03FE		OUT	-1
020A	0DFE		INQ	-1
020B	0A00		ENA	0
020C	02FE	LOOP	INP	-1
020D	581E		RTJ*	ROUT1
020E	0FC4		ALS	4
020F	7C23		SPA*	(HOLD)
0210	02FE		INP	-1
0211	6820		STA*	TEMP
0212	0F42		ARS	2
0213	BC1F		EOR*	(HOLD)
0214	7C1E		SPA*	(HOLD)
0215	D81D		RAO*	HOLD
0216	C81B		LDA*	TEMP
0217	A000		AND	=N$3

P Register	X Register	Mnemonic Instructions		
0218	0003		RTJ*	ROUT1
0219	5812		RTJ*	ROUT1
021A	5811		ALS	2
021B	0FC2		SPA*	(HOLD)
021C	7C16		INP	-1
021D	02FE		STA*	TEMP
021E	6813		ARS	4
021F	0F44		EOR*	(HOLD)
0220	BC12		SPA*	(HOLD)
0221	7C11		RAO*	HOLD
0222	D810		LDA*	TEMP
0223	C80E		AND	=N$F
0224	A000			
0225	000F		RTJ*	ROUT1
0226	5805		RTJ*	ROUT1
0227	5804		SPA*	(HOLD)
0228	7C0A		RAO*	HOLD
0229	D809		JMP*	LOOP
022A	18E1	ROUT1	NOP	0
022B	0B00		ALS	6
022C	0FC6		SPA*	(HOLD)
022D	7C05		INP	-1
022E	02FE		EOR*	(HOLD)
022F	BC03		JMP*	(ROUT1)
0230	1CFA	TEMP	NUM	0
0231	0000	HOLD	NUM	0
0232	0000		END	MTLDR

9. Set the P register.

10. The display should show 232_{16} which means that 52_{10} commands have been entered.

Check Loading Sequence

1. Press master CLEAR switch on console
2. Set all switches to their neutral position
3. Set SELECTIVE STOP switch
4. Set P register
5. Enter into the push button register the first core location to be examined
6. Set X register
7. Set ENTER/SWEEP switch to SWEEP positions
8. Momentarily set the RUN/STEP switch to the STEP position
   The data in the core location specified in step 5 appears on the push button register.
9. To display the next sequential word of core memory in the push button register, briefly set the RUN/STEP switch to the STEP position
10. Release SELECTIVE STOP switch

Execute Bootstrap Loader

1. Mount the MSOS magnetic tape containing the initializer and installable binaries on the magnetic tape unit. Set the unit to equipment 7, unit 0.
2. Set the push button register to xxxx

   xxxx is the length of MAXCOR minus the initializer length (1681_{16})
3. Set the SELECTIVE STOP switch
4. Set the RUN/STEP switch to RUN to load the tape
5. When the tape stops, the system initializer has been read in.

Execute System Initializer

1. Press master CLEAR switch
2. Release SELECTIVE STOP switch
3. Set P register
4. Set the push button register to xxxx which is the address of the system initializer
5. Momentarily set the RUN/STEP switch to RUN
6. SI appears on the teletypewriter to indicate that the system initializer can now load the operating system. Continue with the instructions in part I, section 1.3.5 to install the operating system

1.3.4 LOAD PAPER TAPE VERSION OF SYSTEM INITIALIZER

Mount the disk pack on the disk drive.

Enter the Paper Tape Loading Sequence

Enter the loading sequence into core memory beginning at core memory location 200. This code can be loaded at any location or run anywhere above the last location into which the checksum loader will load, but the location 200 is preferable. This sequence of code will read one formatted record (the checksum loader) into the location specified by the A register (which will be 0000).

1. Press master CLEAR switch
2. Set all switches to neutral positions
3. Set SELECTIVE STOP switch
4. Set P register
5. Set push button register to 200
6. Set ENTER/SWEEP switch to ENTER
7. Set X register
8. Enter the code in this manner:
   a. Enter first (or next word) of code into push button register
   b. Momentarily set RUN/STEP switch to STEP
   c. Press display CLEAR button
   d. Proceed with each word in the X register column using steps a through c until all code is entered (The location is in the P register)

P Register	X Register	Mnemonic Instructions		
0200	6818	START	STA*	STADD
0201	0A20		ENA	$20
0202	E000		LDQ	=X$A1
0203	00A1			
0204	03FE		OUT	-1
0205	0DFE		INQ	-1
0206	02FE	OVER	INP	-1
0207	0111		SAN	1
0208	18FD		JMP*	OVER
0209	0FC8		ALS	8
020A	02FE		INP	-1
020B	680C		STA*	SAVE
020C	0A00	OVER1	ENA	0
020D	02FE		INP	-1
020E	0FC8		ALS	8
020F	02FE		INP	-1
0210	6C08		STA*	(STADD)
0211	D807		RAO*	STADD
0212	C805		LDA*	SAVE
0213	0102		SAZ	2
0214	D803		RAO*	SAVE
0215	18F6		JMP*	OVER1
0216	18FF		NUM	$18FF
0217	0000	SAVE	NUM	0
0218	0000	STADD	NUM	0
			END	START

9. Set the P register
10. The display should show 218_{16} which means that 25_{10} commands have been entered
11. Release SELECTIVE STOP switch

Check Loading Sequence

1. Press master CLEAR switch on console
2. Set all switches their neutral position
3. Set SELECTIVE STOP switch
4. Set P register
5. Enter into the push button register the first core location to be examined
6. Set X register
7. Set ENTER/SWEEP switch to SWEEP position
8. Momentarily set the RUN/STEP switch to the STEP position

   The data in the core location specified in step 5 appears on the push button register
9. To display the next sequential word of core memory in the push button register, briefly set the RUN/STEP switch to the STEP position
10. Release SELECTIVE STOP

Read Checksum Loader

1. Mount the MSOS 3.0 system initializer paper tape on the paper tape reader. The checksum loader is on the front part of this tape
2. Set all switches to neutral
3. Press master CLEAR switch on console
4. Set the P register button
5. Set push button register to 0200
6. Set the RUN/STEP switch to RUN

   The first few feet (checksum loader) are read from the tape into core memory at location 0000, the tape then stops.

Execute Checksum Loader

1. Position the system initialization tape in the paper tape reader
2. Press master CLEAR switch on console
3. Set the A register
4. Set the push button register to xxxx

   xxxx is the length of MAXCOR minus the initializer length (1681_{16})

5. Set the SELECTIVE STOP switch
6. Set the RUN/STEP switch to RUN to load the tape
7. When the tape stops, set the Q register
8. The push button register should be 0000
9. If the push button register does not read 0000, a checksum error occurred
   a. Re-insert the initializer portion of the tape into the reader
   b. Return to step 2

Execute System Initializer

1. Press master CLEAR switch on console
2. Release SELECTIVE STOP switch
3. Set P register
4. Set the push button register to xxxx which is the address of the system initializer
5. Momentarily set the RUN/STEP switch to RUN
6. SI appears on the teletypewriter to indicate that the system initializer can now load the operating system. Continue with the instructions in part I, section 1.3.5 to install the operating system.

1.3.5 INSTALL OPERATING SYSTEM

1. Mount the first MSOS 3.0 installation paper tape in the paper tape reader; ready the unit. Since the initializer and the installable binaries are on the same magnetic tape, if using magnetic tape the tape is already mounted on equipment 7, unit 0. Likewise if using cards, the card deck containing the installable binaries is already in the card reader.
2. Type *S,MAXCOR,xxxx

   xxxx is the highest core location used by the system; use the first column if installing on a 1704 computer and the second column if installing on the 1700 SC.

1704	1700 SC	System Size
3FFF	3FE0	16K
4FFF	4FE0	20K
5FFF	5FE0	24K
6FFF	6FE0	28K
7FFF	7FE0	32K

To reserve areas in upper core for permanent bootstrap loaders and/or core dump programs, set MAXCOR to less than the system core size.

Press RETURN

Message Q

3. Type *S,SECTOR,xxxx

xxxx indicates the maximum number in hexadecimal of disk pack sectors to be used by the operating system.

xxxx	Unit
3E7F	1738-853
7CFF	1738-854
AA9	1751E
1552	1751J
2FFF	For mass memory buffering when using software buffering package

Press RETURN

Message Q

At times it is desirable to limit system scratch by setting SECTOR to a value less than the two maximums mentioned above for the 853 and the 854 disk drives. Reducing the system scratch area provides a file area accessible to the user only.

4. To reduce the size of the core resident system, delete unnecessary drivers at this point in installation.

Type *S,entry point,7FFF

Press RETURN

Message Q

Entry points for the various standard drivers are listed below. Even though only one entry point is listed for each driver, any entry point may be used.

Card	Entry Point
1726-405 card reader	CR405
1728-430 card reader	IN1728
1729-2 card reader	IN1729
Disk	
1738-853/854 disk	DISK
1738-853/854 disk word	DISKWD

Drum	Entry Point
1751 drum	DRMDRZ
Line Printer	
1740-501 line printer	IN501
Magnetic Tape	
1731 unbuffered magnetic tape control	TAPEDR
1731-601-1706 buffered magnetic tape control	TAPDRB
1731 recovery	RECOVT
1731-1706 recovery	RECVTB
1731/1732 tape motion control	T14
1731-601 format ASCII read/write	FRWA
1731-1706-601 buffered format ASCII read/write	FRWAB
1731-601 format binary read/write	FRWB
1731-1706-601 buffered format binary read/write	FRWBB
1731-601 non-format read/write	RWBA
1731-1706-601 buffered non-format read/write	RWBAB
1732-608/609 buffered/unbuffered formatted/unformatted read/write	TAPINT
Paper Tape	
1777 paper tape reader	PTREAD
1777 paper tape punch	PUNCDR
Teletypewriter	
1711/1712 teletypewriter	TYPI
1713 keyboard	S13KI

Examples:

To delete the printer driver:

Type *S, PRINTI, 7FFF

Press RETURN

Message Q

To delete the unbuffered magnetic tape driver, type all non-buffered driver names:

Type *S, TAPEDR, 7FFF

Press RETURN

Message Q

Type *S, FRWA, 7FFF

Press RETURN

Message Q

Type *S, FRWB, 7FFF

Press RETURN

Message Q

Type *S, RECOVT, 7FFF

Press RETURN

Message Q

Type *S, T14, 7FFF

Press RETURN

Message Q

During initialization the printout includes an error 17 message for each of the drivers deleted with an *S.

5. Initializing from other media:

The system initializer is initially set to accept input from a paper tape reader, output to disk, and list on the teletypewriter. A 1711/1712/1713 teletypewriter is assumed to be the comment I/O device.

If the initial input was from the paper tape reader and the operating system is to be built from another device, reassign units at this time. See part III, section 4 for additional initializer control statements and part III, section 6.1 for initializer diagnostics.

To Reassign the Input Device

Type *I, lun

lun	Device
1	1777 paper tape station reader
2	1728-430 or 1729-2 card reader
3	1731-601 or 1732-608 magnetic tape unit (equipment 7, unit 0)
10	1726-405 card reader

Press RETURN

Message Q

To Reassign the Output Device

Type *O, lun

lun	Device
4	1738-853/854 disk pack
5	1751 drum

Press RETURN

Message Q

To Reassign the Comment and List Device

Type *C, lun

lun	Device
6	1711/1712/1713 teletypewriter
7	1740-501 line printer
8	dummy list device

Press RETURN

Message Q

All system initializer messages appear on the comment device with the maps.

6. Type one of the following so that the protect processor can analyze disk I/O requests:

   Type *S, WDADR, 0 if DISK driver is to be used

   *S, WDADR, 1 if DISKWD driver is to be used

   Press RETURN

7. Type *V

   Press RETURN

   Message Q

8. The installable binaries are read by either the card reader, magnetic tape unit, or paper tape reader. The program names on the tape are typed on the list device.

   If using paper tapes, the following message appears after each of the installation tapes is read:

   a. Message L, lun FAILED.

      ACTION

   b. Mount the next installation paper tape on the paper tape reader

   c. Press READY MASTER CLR on the paper tape reader

   d. Type RP

      Press RETURN

During system initialization, the printout is described as follows:

Format: name xxxx

name	The program name
xxxx	First word address (FWA) for core resident (*L) programs
	Beginning sector number of the groups of programs associated with the *YM ordinal for mass memory (*M) resident programs

If unpatched externals result at the end of either an *M load, or an *L load, or at the end of system initialization following an *T command, an ERROR C or ERROR D appears on the system initialization comment device. To continue initialization, repeat the last control statement typed (either *M or *L load commands or the *T.)

a. Type either *M or *L or *T

b. Press RETURN

The list output during initialization is as follows:

```
*S,SYSLVL,5245
*S,DTIMER,7FFF
*S,SYSCOP,7FFF
*S,ONE,7FFF
*S,TWO,7FFF
*S,THREE,7FFF
*S,WDADR,1
*YM,EFILE,1
*YM,LIBEDT,2
*YM,LOADSD,3
*YM,JOBENT,4
*YM,JOBPRO,5
*YM,PROTEC,6
*YM,JPLOAD,7
*YM,JPCHGE,8
*YM,JPT13,9
*YM,MIPRO,10
*YM,RESTOR,11
*YM,ODEBUG,12
*YM,RCOVER,13
*YM,BRKPT,14
*YM,SELF,15
*YM,LOGGER,16
*L    LOCORE
  LOCORE          COPYRIGHT CONTROL DATA CORP. 1970 0000
  SYSBUF          DECEMBER 31, 1970 REV.1           01BB
  TRVEC           DECEMBER 31, 1970 REV.1           06E4
  DRCORE          DECEMBER 31, 1970 REV.1           0710
  NIPROC          DECEMBER 31, 1970 REV.1           0850
```

```
*M          EFILE
  EF                DECEMBER 31, 1970 REV.1          0001
*M          LIBEDT
  LIBEDT            DECEMBER 31, 1970 REV.1          0006
*L     SCHEDU
  SCHEDU            DECEMBER 31, 1970 REV.1          08CE
  NDISP             DECEMBER 31, 1970 REV.1          0978
  NCMPRQ            DECEMBER 31, 1970 REV.1          09B4
  NFNR              DECEMBER 31, 1970 REV.1          09E5
  ADEV              DECEMBER 31, 1970 REV.1          0A4F
*M          LOADSD
  LOAD              DECEMBER 31, 1970 REV.1          0039
  BRANCH            DECEMBER 31, 1970 REV.1          0039
  LIDRIV            DECEMBER 31, 1970 REV.1          0039
  LCDRIV            DECEMBER 31, 1970 REV.1          0039
  LMDRIV            DECEMBER 31, 1970 REV.1          0039
  LLDRIV            DECEMBER 31, 1970 REV.1          0039
  SCAN              DECEMBER 31, 1970 REV.1          0039
  CHPU              DECEMBER 31, 1970 REV.1          0039
  ADJOVF            DECEMBER 31, 1970 REV.1          0039
  CONVRT            DECEMBER 31, 1970 REV.1          0039
  TABSCH            DECEMBER 31, 1970 REV.1          0039
  TABSTR            DECEMBER 31, 1970 REV.1          0039
  LSTOUT            DECEMBER 31, 1970 REV.1          0039
  LINK1             DECEMBER 31, 1970 REV.1          0039
  LINK2             DECEMBER 31, 1970 REV.1          0039
  COREXT            DECEMBER 31, 1970 REV.1          0039
  DPRADD            DECEMBER 31, 1970 REV.1          0039
  LOADER            DECEMBER 31, 1970 REV.1          0039
  NAMPRO            DECEMBER 31, 1970 REV.1          0039
  RBDBZS            DECEMBER 31, 1970 REV.1          0039
  ENTEXT            DECEMBER 31, 1970 REV.1          0039
  XFRPRO            DECEMBER 31, 1970 REV.1          0039
  HEXPRO            DECEMBER 31, 1970 REV.1          0039
  EOLPRO            DECEMBER 31, 1970 REV.1          0039
  ADRPRO            DECEMBER 31, 1970 REV.1          0039
*L     ALCORE
  ALCORE            DECEMBER 31, 1970 REV.1          0C2F
  ALVOL             DECEMBER 31, 1970 REV.1          0CD9
  OFVOL             DECEMBER 31, 1970 REV.1          0CF6
  PARAME            DECEMBER 31, 1970 REV.1          0D02
  COMMON            DECEMBER 31, 1970 REV.1          0D60
  NEPROC            DECEMBER 31, 1970 REV.1          0D77
  NMONI             DECEMBER 31, 1970 REV.1          0DDB
  RW                DECEMBER 31, 1970 REV.1          0E1D
  MAKQ              DECEMBER 31, 1970 REV.1          0EDA
  MINT              DECEMBER 31, 1970 REV.1          0F01
*M          JOBENT
  JOBENT            DECEMBER 31, 1970 REV.1          005A
  T11               DECEMBER 31, 1970 REV.1          005A
  T7                DECEMBER 31, 1970 REV.1          005A
  T5                DECEMBER 31, 1970 REV.1          005A
  T3                DECEMBER 31, 1970 REV.1          005A
```

```
*M          JOBPRO
  JOBPRO          DECEMBER 31, 1970 REV.1          005F
*M          PROTEC
  PROTEC          DECEMBER 31, 1970 REV.1          0064
  JBKILL          DECEMBER 31, 1970 REV.1          0064
*M          JPLOAD
  JPLOAD          DECEMBER 31, 1970 REV.1          006F
*M          JPCHGE
  JPCHGE          DECEMBER 31, 1970 REV.1          0074
  ASCHEX          DECEMBER 31, 1970 REV.1          0074
*M          JPT13
  T13             DECEMBER 31, 1970 REV.1          0078
*M          MIPRO
  MIPRO           DECEMBER 31, 1970 REV.1          007D
*M          RESTOR
  RESTOR          DECEMBER 31, 1970 REV.1          0080
*M          ODEBUG
  ODEBUG          DECEMBER 31, 1970 REV.1          0083
*M          RCOVER
  RCOVER          DECEMBER 31, 1970 REV.1          00A5
  OUTSEL          DECEMBER 31, 1970 REV.1          00A5
  DMPCOR          DECEMBER 31, 1970 REV.1          00A5
  MASDMP          DECEMBER 31, 1970 REV.1          00A5
*M          BRKPT
  BRKPTD          DECEMBER 31, 1970 REV.1          00AE
  BIASCI          DECEMBER 31, 1970 REV.1          00AE
  SIFT            DECEMBER 31, 1970 REV.1          00AE
  RETJMP          DECEMBER 31, 1970 REV.1          00AE
  JUMPTO          DECEMBER 31, 1970 REV.1          00AE
  ENTER           DECEMBER 31, 1970 REV.1          00AE
  ENTCOR          DECEMBER 31, 1970 REV.1          00AE
  PRTREG          DECEMBER 31, 1970 REV.1          00AE
  SETBRP          DECEMBER 31, 1970 REV.1          00AE
  TERMIN          DECEMBER 31, 1970 REV.1          00AE
  DMPCOR          DECEMBER 31, 1970 REV.1          00AE
  MASDMP          DECEMBER 31, 1970 REV.1          00AE
  RESUME          DECEMBER 31, 1970 REV.1          00AE
*M          SELF
  SELFS           DECEMBER 31, 1970 REV.1          00BC
*M          LOGGER
  LOGA            DECEMBER 31, 1970 REV.1          00C1
*L     MASDRV
  MASDRV          DECEMBER 31, 1970 REV.1          0FB7
  S13001          DECEMBER 31, 1970 REV.1          144B
*M     MASS MEMORY DRIVERS
  S13002          DECEMBER 31, 1970 REV.1          00C4
*S,M1713R,S
*M
  S13003          DECEMBER 31, 1970 REV.1          00C7
*S,M1713P,S
*M
  PRT40           DECEMBER 31, 1970 REV.1          00CA
*S,MAS501,S
*M
```

```
  DR1728          DECEMBER 31, 1970 REV.1          00D0
*S,MAS28,S
*M
  CD1729          DECEMBER 31, 1970 REV.1          00DA
*S,MAS292,S
*M
  CR405           DECEMBER 31, 1970 REV.1          00DF
*S,MAS405,S
*M
  DR1732          DECEMBER 31, 1970 REV.1          00E3
*S,MAS32,S
*M
  PUNCDR          DECEMBER 31, 1970 REV.1          00EE
*S,TP1777,S
*M
  PTREAD          DECEMBER 31, 1970 REV.1          00F1
*S,TR1777,S
*L
  TAPEDR          DECEMBER 31, 1970 REV.1          15CA
  FRWA            DECEMBER 31, 1970 REV.1          16E9
  FRWB            DECEMBER 31, 1970 REV.1          17BE
  RWBA            DECEMBER 31, 1970 REV.1          18C1
  RECOVT          DECEMBER 31, 1970 REV.1          1947
  TAPE            DECEMBER 31, 1970 REV.1          19D6
  DISKWD          DECEMBER 31, 1970 REV.1          19E4
  SPACE           DECEMBER 31, 1970 REV.1          1BA1
*S,MAS31,7FFF
*S,TIVALU,7FFF
*S,ABUFAL,7FFF
*S,SNAPE,7FFF
*S,PARITY,7FFF
*S,IPROC1,7FFF
*S,T30,7FFF
*S,T29,7FFF
*S,T28,7FFF
*S,T27,7FFF
*S,T26,7FFF
*S,T25,7FFF
*S,T24,7FFF
*S,T23,7FFF
*S,T22,7FFF
*S,T21,7FFF
*S,T20,7FFF
*S,T19,7FFF
*S,T18,7FFF
*S,T17,7FFF
*S,T16,7FFF
*S,T13,7FFF
*S,T11,7FFF
*S,T8,7FFF
*S,T7,7FFF
*S,T5,7FFF
*S,T3,7FFF
*S,DEBUG,7FFF
*S,TIMACK,7FFF
*F
VRFCTN
*T
```

9. The sector onto which the core image for the new system was written is output on the assigned comment device. The following message appears if there is no timer in the hardware configuration or if there is a timer which rejected.

Message	TIMER RJ	(Indicates that no timer is presently on system)
	SUMMARY LEVEL 01	
	PP	(Set PROGRAM PROTECT switch)

10. Set the PROGRAM PROTECT switch up

11. Type *

    Press RETURN

    Press MANUAL INTERRUPT on the teletypewriter

    Message MI

12. Type *LIBEDT

    Press RETURN

    Message LIB
    IN

13. Type *V,lun

    lun is the logical unit number of the device which contains the input

lun	device
13	card reader
6	magnetic tape unit
2	paper tape uint

    Press RETURN

The following appears on the standard print device:

```
IN

*S,1,0,M
IN

*S,2,3,M
IN

*S,3,0,M
IN

*S,4,1,M
IN

*S,5,2,M
IN

*S,6,3,M
IN

*S,7,2,M
IN

*S,8,2,M
IN

*S,9,2,M
IN

*S,10,2,M
IN

*S,11,2,M
IN

*S,12,3,M
IN

*S,13,3,M
IN

*S,14,3,M
IN

*S,15,3,M
IN

*S,16,3,M
IN

*U
```

The LIBEDT operation fixes the request priorities of the mass resident programs which insures proper allocation of core.

14. To install the SMR (System Maintenance Routines) in the program library:

Type *V,lun

lun	Device
6	magnetic tape
2	paper tape
13	card reader (405)

Press RETURN

The following list appears on the standard print device:

```
IN

*K,I6,P8
IN

*P,F
  DSKTAP  2334
  EQCODE  3150
  MTINP   31BB
  MTOUT   3296
  DCODHX  332D
  CDRIVE  33D2
  MDRIVE  345E
IN

*K,I8
IN

*N,DSKTAP,,,B
IN

*K,I6
IN

*L,DTLP
IN
```

```
*L,UPDATE
I..

*L,LISTR
IN

*L,CYFT
IN

*L,LCOSY
IN

*L,SILP
IN

*L,OPSORT
IN

*K,I6,P8
IN

*P,F
  CONTRL   2334
  LIB      27E4
  IDRIV    2BB4
  MTIDRV   2C13
  PTIDRV   2D07
  CDIDRV   2DB2
  MDRIV    2F0B
  MSDISK   2F13
  MSDRUM   2FC1
  I2       3035
  I2DISK   3045
  I2DRUM   30A6
  I1       310E
  ILOAD    3283
  CDRIV    3926
  LPRINT   39ED
E13

E10
I4
I3
E13

IN

*K,I8
IN

*N,SI,,,B
IN

*U
```

15. To sign off LIBEDT:

    Type *Z

    Press RETURN

    Message J

    The job processor is now in core; normal operations may continue.

16. To verify that MSOS 3.0 is installed:

    Type *P

    Press RETURN

    Message J

17. Type *VRFCTN

    Press RETURN

    Message MSOS 3.0 IS INSTALLED
    J

1.3.6 1700 SC AUTOLOAD PROCEDURES

The last 20_{16} (32_{10}) core locations are protected by the use of the program protect bit in storage and the console AUTO LOAD PROTECT switch. This area is reserved for the AUTOLOAD program. To enter and execute the AUTOLOAD program, follow this sequence.

1. Turn power on
2. Press master CLEAR switch on console
3. Press AUTO LOAD PROTECT switch
4. Set P register
5. Set push button register to 7FE0
6. Set ENTER/SWEEP switch to ENTER
7. Set X register

8. Press display CLEAR button

9. Enter first (or next) word in X register column of AUTOLOAD program; following are the bootstrap instructions:

P Register	X Register	Mnemonic Instructions		
0FE0	C000	SCDKAL	LDA	=N$600
0FE1	0600			
0FE2	6400		STA+	$7FFF
0FE3	7FFF			
0FE4	E000		LDQ	=N$181
0FE5	0181			
0FE6	C000		LDA	=N$102
0FE7	0102			
0FE8	0B00		NOP	0
0FE9	03FE		OUT	-1
0FEA	0D01		INQ	1
0FEB	0A00		ENA	0
0FEC	03FE		OUT	-1
0FED	C000		LDA	=N$7FFF
0FEE	7FFF			
0FEF	0D02		INQ	2
0FF0	03FE		OUT	-1
0FF1	0DFC		INQ	-3
0FF2	0B00		NOP	0
0FF3	02FE	STAT	INP	-1
0FF4	B000		EOR	=N$19
0FF5	0019			
0FF6	0101		SAZ	1
0FF7	18FB		JMP*	STAT
0FF8	1400		JMP+	0
0FF9	0000			

10. Set RUN/STEP switch to STEP position

11. Repeat steps 8 through 10 until the bootstrap routine is entered

12. Return ENTER/SWEEP switch to neutral position

13. Press master CLEAR switch on console

14. Press AUTO LOAD PROTECT switch

MACRO ASSEMBLER 2.0 2

2.1 RELEASE DESCRIPTION

The release description of MSOS 3.0 includes new MSOS features, corrections, deficiencies, and limitations.

2.1.1 NEW FEATURES

Macro Assembler is not updated with the MSOS 3.0 release.

2.1.2 CORRECTIONS

Macro Assembler is not changed or modified with the MSOS 3.0 release.

2.1.3 KNOWN LIMITATIONS

The Macro Assembler punches leader following the paper tape binary output from each program assembled but not preceding any load and go. Although this provides a separator between programs, it does not assure that leader will precede each program, especially the first program.

The assembler does not check for error conditions following completion of a request and thus may process invalid or improper data if the user returns control to the assembler following an I/O error. Unless incorrect data generates assembly diagnostics, disk errors are denoted by MASS MEMORY ERRORS only.

2.1.4 KNOW DEFICIENCIES

A user defined macro which will be used as input to LIBMAC must not contain any images with an * in column 1. A macro which is defined directly within a subprogram may have these images with no restriction.

2.2 REQUIREMENTS

2.2.1 RELEASE MATERIALS

To install Macro Assembler 2.0, the necessary release materials, hardware, and software must be available. These requirements are listed below.

Magnetic Tape Version

One installation magnetic tape containing relocatable programs and control cards

One installation verification deck

Paper Tape Version

One installation paper tape containing relocatable programs and control cards

One installation verification deck

Optional Tapes

One installation paper tape in relocatable format containing library macro preparation programs

One installation paper tape in ASCII format containing system library macros

One COSY source magnetic tape

One list magnetic tape

2.2.2 HARDWARE REQUIREMENTS

The hardware requirements for Macro Assembler 2.0 are the same as for MSOS 3.0 (part I, section 1.2.2).

2.2.3 MEMORY REQUIREMENTS

The largest core load of the assembler, PASS3, requires 3187_{10} plus 260_{10} words of common storage. The remainder of unprotected core is used to build the symbol table. If the length of the symbol table exceeds the length of remaining core, the symbol table is dumped out to mass storage.

2.3 INSTALLATION PROCEDURES

Note the following requirements before installing:

1. MSOS 3.0 must already be installed.
2. Disk is the scratch area for both the Macro Assembler and the load-and-go information.
3. The MSOS parameter SECTOR defines the maximum sector address that the Macro Assembler may use.

The installation instructions are:

1. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

2. If using magnetic tape:

   a. Mount the Macro Assembler release installation magnetic tape on LU 6
   b. When READY lights, type *V,06
   c. Press RETURN

   If using paper tape:

   a. Mount Macro Assembler release installation paper tape on LU 2
   b. Press READY MASTER CLR on paper tape reader
   c. Type *V,02
   d. Press RETURN

   LIBEDT installs the Macro Assembler in the program library and generates the following listing on the list device:

```
IN
*K, I6, P8
IN
*L, ASSEM
IN
```

```
*P, F
     PASS1      nnnn†
     PA1PR2     nnnn†
*K, I8
IN
*N, PASS1, , , B
IN
*K, I6
IN
*P, F
     PASS2      nnnn†
     PA2PR2     nnnn†
IN
*K, I8
IN
*N, PASS2, , , B
IN
*K, I6
IN
*P, F
     PASS3      nnnn†
     PA3PR2     nnnn†
     PA3PR3     nnnn†
IN
*K, I8
IN
*N, PASS3, , , B
IN
*K, I6
IN
*P, F
     PASS4      nnnn†
IN
*K, I8
IN
*N, PASS4, , , B
IN
*K, I6
IN
*N, MACSKL, , , B
IN
*N, MACROS, , , B
IN
*U
```

---

†nnnn load occurs at this address

3. Type *Z

   Press RETURN

   Message J

4. Macro Assembler 2.0 is installed and is ready to assemble source program.

2.4 ADDITIONAL PROCEDURES

2.4.1 SYSTEM MODIFICATION EXAMPLE

The following steps outline the procedures for replacing a file such as PASS1. File name and tape numbers will differ for each system.

1. Reassemble and punch the relocatable information for all programs in the specific pass in binary form (in this case PASS1) so they can be absolutized on the disk. All parts of the pass, in this case PASS1 and PA1PR2, must be present.

2. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

3. Mount relocatable paper tape of PASS 1 on the paper tape reader.
4. Press READY MASTER CLR on the paper tape reader
5. Type *K, I2, P8

   Press RETURN

   Message IN

6. Type *P, F

   Press RETURN

   Message L, 02 FAILED 02
   ACTION

7. Mount the relocatable tape for PA1PR2 on the paper tape reader.
8. Press READY MASTER CLR on the paper tape reader

9. Type RP

   Press RETURN

   The paper tape is read.

   Message L,02 FAILED 02
   ACTION

10. Type CU

    Press RETURN

    Message IN

11. Type *K,I8

    Press RETURN

    Message IN

12. Type *N,PASS1,,,B

    Press RETURN

    Message IN

2.4.2 MODIFICATION OF LIBRARY MACROS EXAMPLE

Use the library macro preparation routine on paper tape 2 to change or add macro definitions to the library macros. Macro Assembler paper tape 2 contains this routine which must be loaded by the 1700 operating system loader.

Input to the program is source macro definitions. Paper tape 3 (system library macros) contains an ENDMAC statement at the end; but the user defined library macros source input tape (s) cannot contain the ENDMAC statement.

Example of library macro preparation:

1. Type *P to load the operating system loader

   Press RETURN

   Message J

2. Mount paper tape 2 (the relocatable binary tape of LIBMAC) on the paper tape reader.

3. Press READY on the paper tape reader

4. Type *L

   Press RETURN

   The paper tape is read

   Message L, 02 FAILED 02
   ACTION

5. Type CU

   Press RETURN

   Message J

6. Mount the paper source tape of user defined macros on the paper tape reader

7. Press READY on the paper tape reader

8. Type *X

   Press RETURN

   Message L, 02 FAILED 02
   ACTION

9. Mount paper source tape 3 (system library macros) on the paper tape reader

10. Press READY on the paper tape reader

11. Type RP

    Press RETURN

    Paper tape 3 is read

    The library macro skeleton permanent file is punched

    Message MACSKL END

12. Remove the new paper tape, NEW MACSKL, from the paper tape punch

13. Press RETURN

    The library macro directory file is punched

    Message J

To insert in the program library macro directory tape which was punched in step 13 and also the NEW MACSKL tape (the library macro skeleton permanent file) which was punched in step 11, use the following steps:

14. Type *LIBEDT

    Press RETURN

    Message LIB

15. Mount paper tape NEW MACSKL which was punched in step 11 on the paper tape reader.

16. Press READY MASTER CLR on the paper tape reader

17. Type *N,MACSKL,,,B

    Press RETURN

    The paper tape NEW MACSKL is read

    Message L, 02 FAILED 02
    ACTION

18. Type CU

    Press RETURN

    Message IN

19. Mount the library macro directory paper tape punched in step 13 on the paper tape reader.

20. Press READY MASTER CLR on the paper tape reader

21. Type *N,MACROS,,,B

    Press RETURN

    The library directory paper tape is read

    Message L, 02 FAILED 02
    ACTION

22. Type CU

    Press RETURN

    Message IN

23. Type *Z

    Press RETURN

    Message J

2.4.3 VERIFICATION OF INSTALLATION

To verify that Macro Assembler 2.0 is installed correctly:

1. Ready the system for operation
2. Press AUTO LOAD on the 1738 disk controller
3. Set the RUN/STEP switch to RUN
4. Message TIMER RJ
   PP
5. Set the PROGRAM PROTECT switch
6. Type *

   Press RETURN
7. Press MANUAL INTERRUPT on the teletypewriter

   Message MI
8. Type *P

   Press RETURN

   Message J
9. Ready the Macro Assembler verification program in the card reader
10. Type *V,11

    Press RETURN

    Message MACRO ASSEMBLER IS INSTALLED
    J

3.1 RELEASE DESCRIPTION

3.1.1 NEW FEATURES

COSY 1.0 has not been changed for the MSOS 3.0 release.

3.1.2 CORRECTIONS

None

3.1.3 KNOWN LIMITATIONS

With standard input assigned to the TTY, COSY still expects an input unit record of 80 characters.

3.1.4 KNOWN DEFICIENCIES

None

3.2 REQUIREMENTS

To install COSY 1.0, the necessary release materials, hardware, and software must be available. These requirements are listed below.

3.2.1 RELEASE MATERIALS

Magnetic Tape Version

One magnetic tape

One installation verification program deck

Paper Tape Version

One paper tape

One installation verification deck

Optional Tapes

One COSY source magnetic tape

3.2.2 HARDWARE REQUIREMENTS

The hardware required to use COSY 1.0 is the same as for the MSOS 3.0 operating system as described in part I, section 1.2.2.

3.2.3 MEMORY REQUIREMENT

COSY 1.0 requires 2829_{10} words of unprotected core to execute.

3.3 INSTALLATION PROCEDURES

MSOS 3.0 must be installed as in part I, section 1.3.

1. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

2. Mount the relocatable binary tape

3. Assign lu to the device which contains the input

4. Type *K,Ilu

   Press RETURN

5. Type *L,COSY

   Press RETURN

   LIBEDT responds when loading is completed

   Message IN

6. Type *Z

   Press RETURN

   Message Q

   COSY is now on the program library and is ready to execute

3.4 ADDITIONAL PROCEDURES

3.4.1 MODIFICATION OF NUMBER OF OUTPUT DEVICES

COSY allows 8 output devices within a single job. To modify the number of COSY output devices, reassemble with the following card changes:

With x as the number of output devices to be used, the TABSIZ card should read

Label	Op	Address
	DEL/	145
TABSIZ	NUM	x

TABLE is a BSS block of 8 words. Delete or add words to allow only enough words for each device to be used. x is the number of output devices to be used.

	DEL/	595
TABLE	BSS	TABLE(x)

3.4.2 VERIFICATION OF INSTALLATION

To verify that COSY is installed:

1. Ready the system for operation
2. Press AUTO LOAD on the 1738 disk controller
3. Set the RUN/STEP switch to RUN
4. Message TIMER RJ

   PP
5. Set the PROGRAM PROTECT switch

6. Type *

   Press RETURN

   Press MANUAL INTERRUPT on the teletypewriter

   Message MI

7. Type *P

   Press RETURN

   Message J

8. COSY verification deck can be used with magnetic tape only. Mount and ready at loadpoint two magnetic tapes on LUN's 6 and 7. Ready the released COSY verification deck in the card reader.

9. Type *V,11

   Press RETURN

   Message COSY IS INSTALLED
   J

MASS STORAGE FORTRAN 2.0A 4

4.1 RELEASE DESCRIPTION

4.1.1 NEW FEATURES

This product is not updated with the MSOS 3.0 release.

4.1.2 CORRECTIONS

None

4.1.3 KNOWN LIMITATIONS

If superfluous information is included on an END line, the program is terminated but no diagnostic is given.

No check is made on the parameter type of the arguments of the intrinsic functions, the external functions, or the statement functions.

4.1.4 KNOWN DEFICIENCIES

The floating point package does not round properly on FW.d format

Runaway diagnostics result if the EQUIVALENCE table overflows. See PSR 528 in Summary 30.

Execution diagnostic 13 is repeated continually. See PSR 529 in Summary 30.

Execution diagnostic 5 is not given, but after writing an ENDFILE succeeding READ and WRITE requests to that unit are ignored. See PSR 527 in Summary 30.

4.2 REQUIREMENTS

To install Mass Storage FORTRAN 2.0A, the necessary hardware and software release materials must be available. These requirements are listed below.

4.2.1 RELEASE MATERIALS

Magnetic Tape Version

One paper tape containing SELCOP and IOCAL

One installation magnetic tape

One installation verification program

Paper Tape Version

One paper tape containing SELCOP and IOCAL

Sixteen installation paper tapes

One installation verification program

Optional Tapes

One COSY source magnetic tape

Three list magnetic tapes

4.2.2 HARDWARE REQUIREMENTS

The minimum hardware configuration is 20K

4.2.3 MEMORY REQUIREMENTS

Mass Storage FORTRAN 2.0A runs in 20K

Instructions	5788
Labeled common (data)	1649
Blank common	1236
Largest overlay	8673

Paper Tape System: Example

Monitor (Disk plus TTY)	7273
1777	503
FORTRAN 2.0A	8673
	14949 words of memory

Card System with Printer: Example

Monitor (Disk plus TTY)	7273
1728-430	891
1740-501	526
FORTRAN 2.0A	8673
	17363 words of memory

Card/Tape System with Printer: Example

Monitor (Disk plus TTY)	7273
1728-430	891
1740-501	526
1731-601 unbuffered	1208
FORTRAN 2.0A	8673
	18571 words of memory

Mixed System: Example

Monitor (Disk plus TTY)	7273
1777	503
1728-430	891
1740-501	526
1731-601 buffered	1311
FORTRAN 2.0A	8673
	19177 words of memory

4.3 INSTALLATION PROCEDURES

MSOS 3.0 must already be installed.

The logical unit numbers must be:

lun 8 for mass storage device

lun 6 for magnetic tape device

lun 2 for paper tape reader with the standard install materials

1. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

2. If using magnetic tape:

   a. Mount the installation magnetic tape on the magnetic tape device

   b. Set the Unit Select Wheel to 0 (LUN 6)

   c. Press LOAD

   d. Press READY

   e. Type *V,06

      Press RETURN

      Message IN

   If using paper tape:

   a. Mount paper tape 1 (Phase A1) on paper tape reader (LUN 2)

   b. Press READY MASTER CLR

   c. Type *V,02

      Press RETURN

      Message IN

   d. Place next tape in paper tape reader

   e. Type *V,02

      Press RETURN

      Message IN

      Continue with step d until all tapes are read.

3. The following output appears on the standard list device during the installation of FORTRAN 2.0A. When paper tape is used, *K,I2,P8 appears instead of *K,I6,P8.

```
IN
*K, I6, P8
IN
*P
     FTN          2991
     GOA          3043
     CNVT         3087
     CONV         30C5
     DIAG         30F8
     EXP9         3188
     FLOAT        3235
     GETSYM       337F
     GPUT         33B8
     IOPRBA       33E1
     PACK         35D1
     Q8PRMS       35F6
```

```
      STORE       3607
      SYMBOL      3635
      LOCLA1      36DC
      DUMYA1      378A
      ENDDO       37F1
      GETC        38F2
      GETF        390B
      GNST        3BDB
      IGETCF      3D7E
      OPTION      3D97
      OUTENT      3DD7
      PHASEA      3E06
      PLABEL      42E6
      Q8QBDS      433C
      RDLABL      433C
      STCHAR      438A
      TYPE        43BC
      ENDLOC      45BC
IN

*K, I8
IN

*N, FORTA1,,, B
IN

*K, I6, P8
IN

*P
      FTN         2991
      GOA         3043
      CNVT        3087
      CONV        30C5
      DIAG        30F8
      EXP9        3188
      FLOAT       3235
      GETSYM      337F
      GPUT        33B8
      IOPRBA      33E1
      PACK        35D1
      Q8PRMS      35F6
      STORE       3607
      SYMBOL      3635
      LOCLA2      36DC
      DUMYA2      378D
      ARITH       379A
      COMNPR      3DE0
      DIMPR       3E76
      GETC        3FFD
```

```
      GETF          4016
      SUBSCR        42E6
      TYPEPR        45A2
      ENDLOC        45B9
IN

*K, I8
IN

*N, FORTA2,,, B
IN

*K, I6, P8
IN

*P
      FTN           2991
      GOA           3043
      CNVT          3087
      CONV          30C5
      DIAG          30F8
      EXP9          3188
      FLOAT         3235
      GETSYM        337F
      GPUT          33B8
      IOPRBA        33E1
      PACK          35D1
      Q8PRMS        35F6
      STORE         3607
      SYMBOL        3635
      LOCLA3        36DC
      DUMYA3        378D
      BYEQPR        379A
      CHECKF        398C
      CONSUB        3A2C
      DATAPR        3AB3
      FGETC         3C43
      FORK          3C62
      GETC          3DD9
      GETF          3DF2
      STCHAR        40C2
      TREE          40F4
      ENDLOC        45F1
IN

*K, I8
IN

*N, FORTA3,,, B
IN
```

```
*K, I6, P8
IN

*P
    FTN         2991
    GOA         3043
    CNVT        3087
    CONV        30C5
    DIAG        30F8
    EXP9        3188
    FLOAT       3235
    GETSYM      337F
    GPUT        33B8
    IORRBA      33E1
    PACK        35D1
    Q8PRMS      35F6
    STORE       3607
    SYMBOL      3635
    LOCLA4      36DC
    DUMYA4      378D
    ARAYSZ      3794
    ASGNPR      37FE
    BDOPR       3844
    CFIVOC      397E
    CKIVC       39DC
    CKNAME      39EC
    CPLOOP      39FC
    ENDDO       3AA1
    GETC        3BA2
    GETF        3BBB
    IOSPR       3E8B
    OUTENT      4519
    RDLABL      4348
    STCHAR      4596
    ENDLOC      45C8
IN

*K, I8
IN

*N, FORTA4, , B
IN

*K, I6, P8
IN

*P
    FTN         2991
    GOA         3042
    CNVT        3087
    CONV        30C5
```

```
    DIAG        30F8
    EXP9        3188
    FLOAT       3235
    GETSYM      337F
    GPUT        33B8
    IOPRBA      33E1
    PACK        35D1
    Q8PRMS      35F6
    STORE       3607
    SYMBOL      3635
    LOCLA5      36DC
    DUMYA5      378A
    ARITH       3797
    GETC        3DE2
    GETF        3DFB
    SUBSCR      40CB
    ENDLOC      4387
IN

*K, I8
IN

*N, FORTA5, , , B
IN

*K, I6, P8
IN

*P
    FTN         2991
    GOA         3043
    CNVT        3087
    CONV        30C5
    DIAG        30F8
    EXP9        3188
    FLOAT       3235
    GETSYM      337F
    GPUT        33B8
    IOPRBA      33E1
    PACK        35D1
    Q8PRMS      35F6
    STORE       3607
    SYMBOL      3635
    LOCLA6      36DC
    DUMYA6      378D
    CFIVOC      3794
    CKIVC       37F2
    ERBPR       3802
    GETC        3855
    GETF        386E
```

```
      MODMXR      3B3E
      RDLABL      3F95
      SUBPPR      3FE3
      TREE        40A3
      ENDLOC      45A0
IN
*K, I8
IN
*N, FORTA6,,, B
IN
*K, I6, P8
IN
*P
      FTN         2991
      GOA         3043
      CNVT        3087
      CONV        30C5
      DIAG        30F8
      EXP9        3188
      FLOAT       3235
      GETSYM      337F
      GPUT        33B8
      IOPRBA      33E1
      PACK        35D1
      Q8PRMS      35F6
      STORE       3607
      SYMBOL      3635
      LOCLA7      36DC
      DUMYA7      378D
      ASEMPR      378D
      EXRLPR      3937
      GETC        3995
      GETF        39AE
      IGETCF      3C7E
      PEQVS       3C97
      PRNTNM      4076
      PUNT        4103
      RDLABL      413B
      SYMSCN      4189
      ENDLOC      41A5
IN
*K, I8
IN
*N, FORTA7,,, B
IN
```

```
*K, I6, P8
IN
*P
     FTN          2991
     GOB          309D
     CNVT         30B5
     DUMMY        30F3
     FCMSTK       31D8
     GETSYM       3261
     IOPRBB       329A
     KCPART       346A
     KOUTPT       349B
     KPCSTK       34AD
     KPC3PR       3868
     KSYMGN       3880
     LABKPC       38C8
     LABLER       38DC
     PUNT         38FA
     Q8PRMS       3910
     STORE        3921
     SYMBOL       394F
     TSALOC       39F2
     LOCLB1       3A7D
     DUMYB1       3B10
     ARAYSZ       3B39
     ASSEM        3BA3
     BANANA       3C0A
     BGINDO       3CCD
     END          3DD6
     ENTCOD       3E1E
     HELEN        3EC7
     INXRST       401E
     NOPROC       4032
     PHASEB       4063
     READIR       44B8
     SUBFUN       4510
     SYMSCN       4577
     ENDLOC       4593
IN
*K, I8
IN
*N, FORTB1, , , B
IN
*K, I6, P8
IN
```

```
*P
     FTN          2991
     GOB          309D
     CNVT         30B5
     DUMMY        30F3
     FCMSTK       31D8
     GETSYM       3261
     IOPRBB       329A
     KCPART       346A
     KOUTPT       349B
     KPCSTK       34AD
     KPC3PR       3868
     KSYMGN       3880
     LABKPC       38C8
     LABLER       38DC
     PUNT         38FA
     Q8PRMS       3910
     STORE        3921
     SYMBOL       394F
     TSALOC       39F2
     LOCLB2       3A7D
     ACP          3B12
     AFIDL        3F58
     ASUPER       3FB2
     CGOTO        4068
     FINK         40C3
     INTRAM       4178
     PARTSB       4351
     SUBPR1       43F3
     SUBPR2       4431
     SUBPR3       44BE
     ENDLOC       4505
IN

*K, I8
IN

*N, FORTB2,,, B
IN

*K, I6, P8
IN

*P
     FTN          2991
     GOB          309D
     CNVT         30B5
     DUMMY        30F3
     FCMSTK       31D8
     GETSYM       3261
```

```
      IOPRBB      329A
      KCPART      346A
      KOUTPT      349B
      KPCSTK      34AD
      KPC3PR      3868
      KSYMGN      3880
      LABKPC      38C8
      LABLER      38DC
      PUNT        38FA
      Q8PRMS      3910
      STORE       3921
      SYMBOL      394F
      TSALOC      39F2
      LOCLB3      3A7D
      ACP         3B10
      ARITHR      3F56
      ASUPER      4114
      FINK        41CA
      INTRAM      427F
      PARTSB      4458
      SUBPR1      44FA
      SUBPR2      4538
      SUBPR3      4505
      ENDLOC      460C
IN

*K, I8
IN

*N, FORTB3, , , B
IN

*K, I6, P8
IN

*P
      FTN         2991
      GOC         3583
      BKDWN       3594
      BLDUP       35F3
      BSS         3636
      CHKWD       3654
      CHOP        37C8
      CL12        39DC
      CON         3A95
      COUNT       3ACC
      DATAST      3AE3
      GETSYM      3B8A
      INOUT       3C2E
      IXOPT       3C9D
```

```
      PHASEC      3DD7
      LABEL       416F
      LABIN       4191
      QXLD        41F7
      REED        4287
      SKIP        42E4
      SYMSCN      433A
      IOPRBC      4356
      Q8PRMS      45E0
      ENDLOC      45F1
IN

*K, I8
IN

*N, FORTC1, , , B
IN

*K, I6, P8
IN

*P
      FTN         2991
      GOOD        2E03
      INDEX       2E28
      IOPRBD      2E44
      NPUNCH      30F4
      Q8PRMS      3230
      PHASE6      3241
      LOCLD1      32E1
      DUMYD1      33A4
      AMT         33B1
      AMOUT       33BA
      ADMAX       39B3
      BKDWN       3BB1
      COUNT       3C1A
      LABOUT      3C31
      NP2OUT      3D10
      RBDX        3D3F
      RBPK        3D7B
      TABDEC      3DA5
      UNPUNC      3E29
      GETSYM      3E3F
      SYMSCN      3E7B
      ENDLOC      3E9D
IN

*K, I8
IN

*N, FORTD1, , , B
IN
```

```
*K, I6, P8
IN

*P
     FTN          2991
     GOOD         2E03
     INDEX        2E28
     IOPRBD       2E44
     NPUNCH       30F4
     Q8PRMS       3230
     PHASE6       3241
     LOCLD2       32E1
     DUMYD2       33A5
     AMT          33AC
     GETSYM       33B3
     IACON        33E1
     IHCON        3439
     NWRITE       3466
     PACK         34A1
     SYMSCN       34CC
     BEGINO       34E8
     FINISH       3694
     ENDLOC       3808
IN

*K, I8
IN

*N, FORTD2,,, B
IN

*K, I6, P8
IN

*P
     FTN          2991
     GOE          2E03
     INDEX        2E28
     IOPRBD       2E44
     NPUNCH       30F4
     Q8PRMS       3230
     PHASE6       3241
     LOCLD1       32E1
     DUMYD1       33A4
     AMT          33B1
     AMOUT        33BA
     ADMAX        398E
     BKDWN        3B8C
     COUNT        3BF5
     LABOUT       3C0C
     NP2OUT       3D1E
```

```
      RBDX        3D56
      RBPK        3D93
      TABDEC      3DBD
      UNPUNC      3E39
      CONV        3E4F
      GETSYM      3E88
      IACON       3ED5
      IHCON       3F2D
      NWRITE      3F59
      PACK        3F94
      SETPRT      3F94
      SYMSCN      4139
      ENDLOC      4155
IN

*K, I8
IN

*N, FORTE1, , , B
IN

*K, I6, P8
IN

*P
      FTN         2991
      GOE         2E03
      INDEX       2E28
      IOPRBD      2E44
      NPUNCH      30F4
      Q8PRMS      3230
      PHASE6      3241
      LOCLD2      32E1
      DUMYD2      33A5
      AMT         33AC
      CONV        33B3
      GETSYM      33EC
      IACON       3439
      IHCON       3491
      NWRITE      34BD
      PACK        34F8
      SETPRT      3523
      SYMSCN      3699
      BEGINO      36B5
      FINISH      37FE
      ENDLOC      396D
IN

*K, I8
IN
```

```
*N, FORTE2,,, B
IN

*K, I6, P8
IN

*L, FTN
IN

*L, Q8IFRM
IN

*L, Q8FS
IN

*L, Q8TRAN
IN

*L, FLOT
IN

*L, Q8QINI
IN

*L, Q8QEND
IN

*L, Q8CMP1
IN

*L, Q8RWBU
IN

*L, Q8ERRM
IN

*L, Q8DFNF
IN

*L, Q8QX
IN

*L, Q8QUN1
IN

*L, Q8FGET
IN

*L, Q8MAGT
IN

*L, Q8QBCK
IN

*L, IOCK
IN
```

*L, Q8PSE
IN

*L, Q8PAND
IN

*L, Q8EXP9
IN

*L, Q8EXP1
IN

*L, Q8AB
IN

*L, SIGN
IN

*L, EXP
IN

*L, SQRT
IN

*L, ALOG
IN

*L, TANH
IN

*L, SIN
IN

*L, ATAN
IN

*L, QSAVE
IN

*L, IFALT
IN

*L, Q8FX
IN

*L, Q8PREP
IN

*U

Type: *Z

Press: CARRIAGE RETURN

Message: J

4.4 ADDITIONAL PROCEDURES

4.4.1 LOADING AND CALLING SELCOP

SELCOP is a utility program helpful in building a 1700 FORTRAN installation tape or deck. It consists of two programs: SELCOP and IOCAL. IOCAL handles the I/O for SELCOP.

The SELCOP program allows an operator to build a tape from either a tape or a deck of relocatable binary programs. SELCOP may:

Select a binary relocatable program from the equipped input unit and copy the program on the equipped output unit

Change equipped units during program execution

Rewind any tape drive

Transfer a number of records from one unit to another without using the system standard units

Loading SELCOP and IOCAL into Program Library

1. MSOS 3.0 must be installed.

2. Type *P

   Press RETURN

   Message J

3. Load the SELCOP paper tape into the paper tape reader

   Press READY MASTER CLR on the reader

4. Type *K,I2

   Press RETURN

   Message J

5. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

6. Type *L,SELCOP

   Press RETURN

   Part of the paper tape is read.

   Message IN

7. Type *L,IOCAL

   Press RETURN

   The rest of the paper tape is read.

   Message IN

Calling SELCOP

To call SELCOP, the object library must be installed.

1. Type *P

   Press RETURN

2. Type SELCOP

   Press RETURN

   Message IN

3. Type one of the five commands listed under SELCOP commands

4. Message NEXT

5. Type another of the five commands if desired

6. Since SELCOP is written in 1700 FORTRAN, any errors of incorrect input of logical unit formats will result in a FORTRAN I/O ERROR, and any errors in number of logical units will result in a 1700 MSOS J02 error. When the program terminates because of errors, recall SELCOP and check correct formats and logical unit numbers for the installation.

SELCOP Commands

Equipping Command (*K): The *K statement must precede the *N command. Its function is to equip the input unit and the punch unit. This command only affects SELCOP; it does not affect MSOS.

Type *K

Press RETURN

Type lu,lu
lu,lu are the two-digit parameters of logical units to be equipped.

Example:

02,03 equips the paper tape reader as input, paper tape punch as output.

Transfer Command (*T): The *T statement is used to transfer a number of records from one unit to another. The I/O consists of formatted binary reads and writes.

Type *T

Press RETURN

Type lu,lu,xxxx
lu,lu are the 2 two-digit parameters of logical units to be equipped.

xxxx is a four-digit record count.

Since the reads from input comments are in FORTRAN, it is important to use 2-digit logical unit numbers and 4-digit record counts.

Example:

02,06,0030 tells SELCOP to transfer thirty records from logical unit 2 to logical unit 6.

Name Command (*N): The *N is the main SELCOP command. When SELCOP receives a six-character name, it searches the input unit which was equipped by the *K statement to find a 2050_{16} NAM block. If that NAM block is the name which was input from the teletypewriter, SELCOP copies the program on output unit until it finds a $C050_{16}$ XFR block which terminates the command.

Type *N

Press RETURN

Type a 6-character ASCII name (if the name is fewer than 6 characters, right fill to 6 characters with blanks).

Example:

VERIFY will find program VERIFY on the input unit and copy it onto the output unit.

Rewind Command (*R): Use the *R statement to rewind a specific magnetic tape drive.

Type *R

Press RETURN

Type one two-digit logical unit number

Example:

Typing 07 rewinds tape drive 1 which is logical unit 7.

Stop Command (*S):

Type *S

Press RETURN

Message J (this signifies entry into the system; SELCOP terminates)

4.4.2 BUILDING A MASS STORAGE FORTRAN 2.0A OR 2.0B INSTALLATION TAPE

Requirements

To build a Mass Storage FORTRAN installation tape, the following requirements must first be met:

Installation of MSOS 3.0

Installation of Macro Assembler 2.0

Installation of SELCOP

Installation of FORTRAN 2.0A or 2.0B

20K memory for FORTRAN 2.0A; 24K for 2.0B

Either:

2 magnetic tape units, or

1 magnetic tape unit; 1 paper tape reader; and 1 paper tape punch

Procedures

1. Compile or assemble all of the source tape programs using the P otion to punch on either magnetic or paper tape. Sections 7.4.1 (2.0A) and 7.5.1 (2.0B) list installation tape contents (part III).
2. Call SELCOP using the procedures outlined in 4.4.1.
3. Use SELCOP's *T statement as described in 4.4.1 to transfer the control characters from the teletype to the tape being constructed.

   Type *T

   Press RETURN

   Type lu,lu,0002

   Press RETURN

   For magnetic tape:

   Type *K,I6,P8

   Press LINE FEED

   Press RETURN

   Type *P

   Press RETURN

   For paper tape:

   Type *K,I2,P8

   Press LINE FEED

   Press RETURN

   Type *P

   Press RETURN

4. Use the *N statement (4.4.1) to select and copy the programs from the binaries generated in step 1 onto the tape being generated, beginning with FTN and continuing in the order given in the tape contents section as: GOA, CNVT, ..., ENDLOC. Note that the logical units must be equipped with the *K statement before the *N statement is used.

5. Use the *T statement (4.4.1) to transfer the control characters. Then:

   Type *T

   Press RETURN

   Type lu,lu,0003

   Press RETURN

   Type *T

   Press LINE FEED

   Press RETURN

   Type *K,I8

   Press LINE FEED

   Press RETURN

   Type *N, FORTxx,,,B
   xx is the last record in the phase being built

   Press LINE FEED

   Press RETURN

6. If paper tape is being used, it is suggested that the following be typed at the end of each phase:

   Type *U

   Press LINE FEED

   Press RETURN

7. Repeat steps 2 through 6 for each phase.

4.4.3 CONSTRUCTION OF OBJECT LIBRARY

Use the following steps for each object library:

1. Call SELCOP as outlined in 4.4.1.

2. Use the *T statement (4.4.1) to transfer the control character from the teletype to the tape being constructed.

   Type *T

   Press RETURN

   Type lu,lu,0001

   Press RETURN

   Type *L, entry point name
   Part 7.4.5 lists entry point names for 2.0A
   Part 7.5.5 lists entry point names for 2.0B

   Press RETURN

3. Use the *N statement (4.4.1) to select and copy the program from the tape of binaries onto the tape being generated.

4. Repeat steps 2 and 3 for each specified entry point.

5. After entering the last program in the program library, transfer an *U onto the tape being generated.

   Type *T

   Press RETURN

   Type lu,lu,0001

   Press RETURN

   Type *U

   Press RETURN

4.4.4 PHASE MODIFICATION

1. Compile and/or assemble all programs which appear in the phase to be modified. The relocatable output must be put in absolute form according to the order specified in the FORTRAN 2.0A or 2.0B tape formats in part 7.4.3 or part 7.5.3 to establish a relocatable tape of the programs in the modified phase.

2. Type *P

   Press RETURN

   Message J

3. Type *LIBEDT

   Press RETURN

   Message LIB
           IN

4. If input is from paper tape:

   Type *K,I2,P8

   Press RETURN

   If input is from magnetic tape:

   Type *K,I6,P8

   Press RETURN

5. Type *P

   Press RETURN

   Action The system reads the tape

   Message IN (if there is an *T at the end of the tape)

6. Type *K,I8

   Press RETURN

   Message IN

7. Type *N, file name of modified phase,,,B

   Part 7.4.3 lists FORTRAN 2.0A phase names
   Part 7.5.3 lists 2.0B phase names

   Press RETURN

   Message IN

4.4.5 OBJECT LIBRARY MODIFICATION

1. When a subroutine in the object time library needs to be changed, assemble or compile the routine on a relocatable tape.

2. Type *P

   Press RETURN

   Message J

3. Type *LIBEDT

   Press RETURN

   Message LIB
           IN

4. Type *L, routine entry point name

   Entry point names are in 7.4.5. They are the same for 2.0A and 2.0B

   Press RETURN

   Message IN

4.4.6 VERIFICATION OF INSTALLATION

1. Ready the system for operation
2. Press AUTO LOAD on the 1738 disk controller
3. Set the RUN/STEP switch to RUN
4. Message TIMER RJ
           PP
5. Set the PROGRAM PROTECT switch
6. Type *

   Press RETURN

   Press the MANUAL INTERRUPT button on the teletypewriter

   Message MI

7. Type *P

   Press RETURN

   Message J

8. Ready the released FORTRAN verification deck in the card reader; ready the unit
9. Type *V, 11

   Press RETURN

   Message OPTIONS

10. Type LX

    Press RETURN

    Message FORTRAN IS INSTALLED
            J

5.1 RELEASE DESCRIPTION

5.1.1 NEW FEATURES

This product is not updated with the MSOS 3.0 release

5.1.2 CORRECTIONS

None

5.1.3 KNOWN LIMITATIONS

If superfluous information is included on an END line, the program is terminated but no diagnostic is given.

No check is made on the parameter type of the arguments of the intrinsic functions, the external functions, or the statement functions.

5.1.4 KNOWN DEFICIENCIES

The floating point package does not round properly on FW.d format.

Runaway diagnostics result if the EQUIVALENCE table overflows. See PSR 528 in Summary 30.

Execution diagnostic 13 is repeated continually. See PSR 529 in Summary 30.

Execution diagnostic 5 is not given, but after writing an ENDFILE, succeeding READ and WRITE requests to that unit are ignored. See PSR 527 in Summary 30.

5.2 REQUIREMENTS

To install Mass Storage FORTRAN 2.0B, the necessary release materials, hardware, and software must be available. These requirements are listed below.

5.2.1 RELEASE MATERIALS

Magnetic Tape Version

One paper tape containing SELCOP and IOCAL

One installation magnetic tape

One installation verification program

Paper Tape Version

One paper tape containing SELCOP and IOCAL

Ten installation paper tapes

One installation verification program

Optional Tapes

One COSY source magnetic tape

Three list magnetic tapes

5.2.2 HARDWARE REQUIREMENTS

The minimum hardware configuration is 20K.

5.2.3 MEMORY REQUIREMENTS

Version 2.0B is designed to run in 24K.

Instructions	9800
Labeled common (data)	1649
Blank common	1236
Largest overlay	12,685

5.3 INSTALLATION PROCEDURES

MSOS 3.0 must already be installed.

The logical unit numbers must be:

lun 8 for mass storage device

lun 6 for magnetic tape device

lun 2 for paper tape reader with the standard install materials

1. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

2. If using magnetic tape:

   a. Mount the installation magnetic tape on the magnetic tape device

   b. Set the unit select wheel to 0 (LUN 6)

   c. Press LOAD

   d. Press READY

   e. Type *V,06

      Press RETURN

      Message IN

   If using paper tape:

   a. Mount paper tape 1 (phase A1) on paper tape reader (LUN 2)

   b. Press READY MASTER CLR

   c. Type *V,02

      Press RETURN

      Message IN

   d. Place next tape in paper tape reader

   e. Press READY MASTER CLR

   f. Type *V,02

      Press RETURN

      Continue with step d until all tapes are read.

3. The following output appears on the standard list device during the installation of FORTRAN 2.0B. When paper tape is used, *K,I2,P8 appears instead of *K,I6,P8.

```
IN
*K, I6, P8
IN
*P
      FTN          25EA
      GOA          2CAO
      CFIVOC       23E4
      CKNAME       2D42
      CNVT         2D52
      CONV         2D90
      DIAG         2DC3
      EXP9         2E53
      FLOAT        2F00
      GETC         304A
      GETF         3075
      GETSYM       336D
      GPUT         33A6
      IGETCF       33CF
      IOPRBA       33E8
      PACK         3684
      Q8PRMS       36A9
      RDLABL       36BA
      STORE        3708
      SYMBOL       3736
      ENDDO        37DD
      GNST         38DE
      OPTION       3A81
      OUTENT       3AC1
      PHASEA       3AF5
      PLABEL       3FD4
      STCHAR       402A
      TYPE         405C
      LOCLA1       4259
      DUMYA1       4316
      Q8QBDS       437D
      ENDLOC       437D
IN
*K, I8
IN
*N, FORTA1, , , B
IN
*K, I6, P8
IN
```

```
*P
     FTN          25EA
     GOA          2CAO
     CFIVOC       2CE4
     CKNAME       2D42
     CNVT         2D52
     CONV         2D90
     DIAG         2DC3
     EXP9         2E53
     FLOAT        2F00
     GETC         304A
     GETF         3075
     GETSYM       336D
     GPUT         33A6
     IGETCF       33CF
     IOPRBA       33E8
     PACK         3684
     Q8PRMS       36A9
     RDLABL       36BA
     STORE        3708
     SYMBOL       3736
     ENDDO        37DD
     GNST         38DE
     OPTION       3A81
     OUTENT       3AC1
     PHASEA       3AF5
     PLABEL       3FD4
     STCHAR       402A
     TYPE         405C
     LOCLA2       4259
     DUMYA2       4316
     BYEQPR       437D
     CHECKF       456F
     COMNPR       460F
     CONSUB       46A5
     DATAPR       472C
     DIMPR        48BC
     EXRLPR       4A43
     FGETC        4AA1
     FORK         4AC0
     PEQVS        4C37
     PRNTNM       5016
     SUBPPR       50A3
     SYMSCN       5163
     TYPEPR       517F
     ENDLOC       5196
IN
```

```
*K, I8
IN

*N, FORTA2,,, B
IN

*K, I6, P8
IN

*P

     FTN          25EA
     GOA          2CA0
     CFIVOC       2CE4
     CKNAME       2D42
     CNVT         2D52
     CONV         2D90
     DIAG         2DC3
     EXP9         2E53
     FLOAT        2F00
     GETC         304A
     GETF         3075
     GETSYM       336D
     GPUT         33A6
     IGETCF       33CF
     IOPRBA       33E8
     PACK         3684
     Q8PRMS       36A9
     RDLABL       36BA
     STORE        3708
     SYMBOL       3736
     ENDDO        37DD
     GNST         38DE
     OPTION       3A81
     OUTENT       3AC1
     PHASEA       3AF5
     PLABEL       3FD4
     STCHAR       402A
     TYPE         405C
     LOCLA3       4259
     DUMYA3       4316
     ARAYSZ       437D
     ASEMPR       43E7
     ASGNPR       4591
     BDOPR        45D7
     CHECKF       4711
     CKIVC        47B1
     CONSUB       47C1
     CPLOOP       4848
     DATAPR       48ED
```

```
      FGETC       4A7D
      FORK        4A9C
      ERBPR       4C13
      MODMXR      4C66
      PUNT        50BD
      ENDLOC      50F5
IN

*K, I8
IN

*N, FORTA3, , , B
IN

*K, I6, P8
IN

*P

      FTN         25EA
      GOA         2CA0
      CFIVOC      2CEA
      CKNAME      2D42
      CNUT        2D52
      CONV        2D90
      DIAG        2DC3
      EXP9        2E53
      FLOAT       2F00
      GETC        304A
      GETF        3075
      GETSYM      336D
      GPUT        33A6
      IGETCF      33CF
      IOPRBA      33E8
      PACK        3684
      Q8PRMS      36A9
      RDLABL      36BA
      STORE       3708
      SYMBOL      3736
      ENDDO       37DD
      GNST        38DE
      OPTION      3A81
      OUTENT      3AC1
      PHASEA      3AF5
      PLABEL      3FD4
      STCHAR      402A
      TYPE        405C
      LOCLA4      4259
      DUMYA4      4316
      ARITH       437D
      SUBSCR      49EF
```

```
      TREE        4CAD
      ENDLOC      51A1
IN

*K, I8
IN

*N, FORTA4, , , B
IN

*K, I6, P8
IN

*P

      FTN         25EA
      GOA         2CA0
      CFIVOC      2CE4
      CKNAME      2D42
      CNVT        2D52
      CONV        2D90
      DIAG        2DC3
      EXP9        2E53
      FLOAT       2F00
      GETC        304A
      GETF        3075
      GETSYM      336D
      GPUT        33A6
      IGETCF      33CF
      IOPRBA      33E8
      PACK        3684
      Q8PRMS      36A9
      RDLABL      36BA
      STORE       3708
      SYMBOL      3736
      ENDDO       37DD
      GNST        38DE
      OPTION      3A81
      OUTENT      3AC1
      PHASEA      3AF5
      PLABEL      3FD4
      STCHAR      402A
      TYPE        405C
      LOCLA5      4259
      DUMYA5      4316
      BDOPR       437D
      CKIVC       44B7
      IOSPR       44C7
      ENDLOC      4B69
IN
```

```
*K, I8
IN

*N, FORTA5, , , B
IN

*K, I6, P8
IN

*P
     FTN          25EA
     GOB          2CFA
     CNVT         2DIO
     DUMMY        2D4E
     FCMSTK       2E33
     GETSYM       2EBC
     IOPRBB       2EF5
     KCPART       347D
     KOUTPT       34AE
     KPCSTK       34C0
     KPC3PR       387B
     KSYMGN       3893
     LABKPC       38DB
     LABLER       38EF
     PUNT         390D
     Q8PRMS       3923
     STORE        3934
     SYMBOL       3962
     TSALOC       3A05
     ARAYSZ       3A90
     ASSEM        3AFA
     BANANA       3B61
     BGINDO       3C24
     END          3D2D
     ENTCOD       3D75
     HELEN        3E1E
     INXRST       3F75
     NOPROC       3F89
     PHASEB       3FBA
     READIR       440F
     SUBFUN       4467
     SYMSCN       44CE
     ACP          44EA
     AFIDL        4930
     ASUPER       498A
     CGOTO        4A40
     FINK         4A9B
     INTRAM       4B50
     PARTSB       4D29
     SUBPR1       4DCB
```

```
      SUBPR2      4E09
      SUBPR3      4E96
      ARITHR      4EDD
      ENDLOC      509B
IN

*K, I8
IN

*N, FORTB1, , , B
IN

*K, I6, P8
IN

*P

      FTN         25EA
      GOC         31E0
      BKDWN       31F8
      BLDUP       3257
      BSS         329A
      CHKWD       32B8
      CHOP        342C
      CL12        3640
      CON         36F9
      COUNT       3730
      DATAST      3747
      GETSYM      37EE
      INOUT       3892
      IOPRBC      3901
      IXOPT       47AC
      LABEL       48E6
      LABIN       4908
      PHASEC      496E
      Q8PRMS      4CB8
      QXLD        4CC9
      REED        4D59
      SKIP        4DB6
      SYMSCN      4E0C
      ENDLOC
IN

*K, I8
IN

*N, FORTC1, , , B
IN

*K, I6, P8
IN
```

```
*P
     FTN          25EA
     GOOD         31E0
     AMOUT        3203
     ADMAX        37C8
     BEGINO       39C6
     BKDWN        3AD6
     COUNT        3B3F
     FINISH       3B56
     GETSYM       3CC5
     IACON        3D69
     IHCON        3DC1
     INDEX        3DEE
     IOPRBD       3E0A
     LABOUT       43B7
     NP2OUT       4496
     NPUNCH       44C5
     NWRITE       4601
     PACK         463C
     PHASE6       4667
     Q8PRMS       4703
     RBDX         4714
     RBPK         4750
     SYMSCN       477A
     TABDEC       4796
     UNPUNC       481A
     ENDLOC       4830
IN

*K, I8
IN

*N, FORTD1,,, B
IN

*K, I6, P8
IN

*P
     FTN          25EA
     GOE          31E0
     AMOUT        3203
     ADMAX        37D7
     BEGINO       39D5
     BKDWN        3B16
     CONV         3B7F
     COUNT        3BB8
     FINISH       3BCF
     GETSYM       3D3E
     IACON        3DE2
```

```
*L, Q8ERRM
IN
*L, Q8DFNF
IN
*L, Q8QX
IN
*L, Q8QUN1
IN
*L, Q8FGET
IN
*L, Q8MAGT
IN
*L, Q8QBCK
IN
*L, IOCK
IN
*L, Q8PSE
IN
*L, Q8PAND
IN
*L, Q8EXP9
IN
*L, Q8EXP1
IN
*L, Q8AB
IN
*L, SIGN
IN
*L, EXP
IN
*L, SQRT
IN
*L, ALOG
IN
*L, TANH
IN
*L, SIN
IN
```

```
      IHCON       3E3A
      INDEX       3E66
      IOPRBD      3E82
      LABOUT      442F
      NP2OUT      4541
      NPUNCH      4579
      NWRITE      46B5
      PACK        46F0
      PHASE6      471B
      Q8PRMS      47B7
      RBDX        47C8
      RBPK        4805
      SETPRT      482F
      SYMSCN      49A9
      TABDEC      49C5
      UNPUNC      4A41
      ENDLOC      4A57
IN

*K, I8
IN

*N, FORTE1, , , B
IN

*K, I6, P8
IN

*L, FTN
IN

*L, Q8IFRM
IN

*L, Q8FS
IN

*L, Q8TRAN
IN

*L, FLOT
IN

*L, Q8QINI
IN

*L, Q8QEND
IN

*L, Q8CMP1
IN

*L, Q8RWBU
IN
```

*L, ATAN
IN

*L, QSAVE
IN

*L, IFALT
IN

*L, Q8FX
IN

*L, Q8PREP
IN

*U

Type: *Z

Press: RETURN

Message: J

5.4 ADDITIONAL PROCEDURES

See the additional procedures described under the Mass Storage FORTRAN 2.0A additional procedures in part I, section 4.4. These procedures include:

	Section
Loading and calling SELCOP	4.4.1
Building a Mass Storage FORTRAN 2.0A or 2.0B Installation tape	4.4.2
Construction of object library	4.4.3
Phase modification	4.4.4
Object library modification	4.4.5
Verification	4.4.6

SYSTEM CHECKOUT 1.0 6

6.1 RELEASE DESCRIPTION

6.1.1 NEW FEATURES

The 1.0 version of System Checkout is the first release; therefore, there are no new features for the 3.0 release of MSOS.

6.1.2 CORRECTIONS

None

6.1.3 KNOWN LIMITATIONS

The following module must be modified to use System Checkout with MSOS 3.0:

Label	Op	Address
CO3RD	DCK/	I=lu H=lu
	DEL/	87, 88
	EXT	EFILE, LOADSD, JOBENT, JOBPRO, PROTEC
	EXT	JPLOAD, JPCHGE, JPT13, LIBEDT
	EXT	RESTOR, RCOVER, BRKPT, SELF, LOGGER
	DEL/	488
AC3JBT	ADC	EFILE, LOADSD
	DEL/	492
	ADC	JOBPRO, JPLOAD, JPCHGE, JPT13
	ADC	RESTOR
	DEL/	494
	ADC	PROTEC, LIBEDT, RCOVER, BRKPT, SELF
	ADC	LOGGER

6.1.4 KNOWN DEFICIENCIES

None

6.2 REQUIREMENTS

6.2.1 RELEASE MATERIALS

Magnetic Tape Version

One installation tape

Paper Tape Version

Six installation tapes

Optional Tapes

One COSY source magnetic tape

One list magnetic tape

6.2.2 HARDWARE REQUIREMENTS

The minimum hardware configuration is the same as for MSOS 3.0, listed in part I, section 1.2.2.

6.2.3 MEMORY REQUIREMENTS

System Checkout 1.0 uses no more than 500 words of core memory.

6.3 INSTALLATION PROCEDURES

6.3.1 LOADING DURING INITIALIZATION

1. Assign the next two available mass memory system directory ordinals to SYSCOP and SYSSEG (under which the System Checkout programs will be loaded). As an example, 19 and 20 are used as system ordinals for SYSCOP and SYSSEG in this section.

```
*YM,SYSCOP,19,SYSSEG,20
```

2. Remove the *S, SYSCOP, 7FFF statement from the beginning of tape.

3. Load the SYSCOP program after the 19th *M; load CO1ST, CO2ND, CO3RD, and COLAST after the 20th *M.

4. Set COBOPS and/or COBOPL as entry points.

   COBOPS is the starting sector of a block of mass memory on which the COBOP program will write the failed image. If unpatched, COBOPS is assumed to be $3D29 which allows a 32K image to be written on the highest sectors of an 853 disk. Because this block may be part of scratch, do not use scratch while SYSCOP is running or if the image is to be saved.

   COBOPL is the length of transfer by COBOP. Set it according to the core size of the system.

System Size	COBOPL
16K	$3FFF
20K	$4FFF
24K	$5FFF
28K	$6FFF
32K	$7FFF assumed if unpatched

   Set COBOPS and/or COBOPL entry points in one of the following ways.

   Use a core resident program (*L program) which was loaded prior to SYSCOP, CO1ST, CO2ND, CO3RD, and COLAST,

   or

   use the *S statement. Using the *S statement, the following example reflects a 28K system with the failed image to be written starting at sector $2000:

```
*S, COBOPL, 6FFF
*S, COBOPS, 2000
```

5. Load COBOP using the following information:

   COBOP must be loaded under an *L (core resident program) before loading CO1ST, CO2ND, CO3RD, and COLAST.

   COBOP requires $3A of core memory.

   COBOP can only be used with an 853/854 disk assigned to equipment code 3 on the A-Q channel.

   Externals COBOPS and COBOPL may be unpatched.

6. Load SYSCOP using the following information:

   Load SYSCOP under the SYSCOP ordinal.

   SYSCOP requires $17E locations of allocatable core during execution.

   COBOPS may be unpatched.

   Schedule the SYSCOP ordinal at priority level 3 (MIPRO may be used) in step 7 of 6.4.1.

7. Load CO1ST, CO2ND, CO3RD, and COLAST considering the following:

   Load CO1ST through COLAST under the SYSSEG ordinal so that CO1ST is first and COLAST is last.

   FMASK and/or NDISP may or may not be unpatched, depending on the configuration.

   Load SYSCOP after the following programs:

   TRVEC
   NIPROC
   NMONI
   NFNR
   DRCORE
   PARAME
   ALVOL
   NCMPRQ
   NDISP or RDISP
   COMMON
   MINT
   COBOP

8. The sample format for inserting the information specified in steps 1 through 7 is:

```
SI
*Y . . .
.
.
.
*YM. . .
.
.
.
*YM,SYSCOP,19,SYSSEG,20
.
.
.
*S,COBOPS,2000
*S,COBOPL,6FFF
.
.
.
*L
     LOCORE      0000
.
.                                                    (Monitor Modules)
.
*L
.
.
.
     COBOP       hhhh
.
.
.
*M               (19th *M)
     SYSCOP      hhhh
*M               (20th *M)
     CO1ST       hhhh
     CO2ND       hhhh
     CO3RD       hhhh
     COLAST      hhhh
.
.
.
*T
```

9. After loading, use LIBEDT to set the system request priorities to four as in the following example based on the information in step 8.

```
Message J
Type *LIBEDT
Press RETURN
Message LIB
        IN
Type *S,19,4,M
Press RETURN
Message IN
Type *S,20,4,M
Press RETURN
Message IN
```

Sample System Initializer Typeout

```
SI
*S,MAXCOR,6FFF
Q
*S,SECTOR,3E7F
Q
*S,COBOPL, 6FFF
Q
*S,COBOPS,2000
Q
*I,3
Q
*C,7
Q
*V
ERROR    C
Q
*L
TIMER RJ
PP
*
MI
*LIBEDT
    LIB
    IN
*V,6
    IN
```

```
TIMER RJ
PP
*
TIMER RJ
PP
*
MI
=S019,3
SELECT OPTION
```

System Initialization Typeout

```
*S,COBOPS,2000
*S,COBOPL,6FFF
*S,SYSLVL,5245
*S,DTIMER,7FFF
*S,ONE,7FFF
*S,TWO,7FFF
*S,THREE,7FFF
*S,WDADR,1
*YM,EFILE,1
*YM,LIBEDT,2
*YM,LOADSD,3
*YM,JOBENT,4
*YM,JOBPRO,5
*YM,PROTEC,6
*YM,JPLOAD,7
*YM,JPCHGE,8
*YM,JPT13,9
*YM,MIPRO,10
*YM,RESTOR,11
*YM,ODEBUG,12
*YM,RCOVER,13
*YM,BRKPT,14
*YM,SELF,15
*YM,LOGGER,16
*YM,SYSCOP,17,SYSSEG,18
*L    LOCORE
  LOCORE          COPYRIGHT CONTROL DATA CORP. 1970 0000
  SYSBUF          DECEMBER 31, 1970 REV.1           01C9
  TRVEC           DECEMBER 31, 1970 REV.1           06F2
  DRCORE          DECEMBER 31, 1970 REV.1           071E
  NIPROC          DECEMBER 31, 1970 REV.1           085E
*M        EFILE
  EF              DECEMBER 31, 1970 REV.1           0001
*M        LIBEDT
  LIBEDT          DECEMBER 31, 1970 REV.1           0006
*L    SCHEDU
  SCHEDU          DECEMBER 31, 1970 REV.1           08DC
  NDISP           DECEMBER 31, 1970 REV.1           0986
  NCMPRQ          DECEMBER 31, 1970 REV.1           09C2
  NFNR            DECEMBER 31, 1970 REV.1           09F3
  ADEV            DECEMBER 31, 1970 REV.1           0A5D
  COBOP           COPYRIGHT CONTROL DATA CORP. 1970 0C3D
```

```
*M        LOADSD
  LOAD          DECEMBER 31, 1970 REV.1          0039
  BRANCH        DECEMBER 31, 1970 REV.1          0039
  LIDRIV        DECEMBER 31, 1970 REV.1          0039
  LCDRIV        DECEMBER 31, 1970 REV.1          0039
  LMDRIV        DECEMBER 31, 1970 REV.1          0039
  LLDRIV        DECEMBER 31, 1970 REV.1          0039
  SCAN          DECEMBER 31, 1970 REV.1          0039
  CHPU          DECEMBER 31, 1970 REV.1          0039
  ADJOVF        DECEMBER 31, 1970 REV.1          0039
  CONVRT        DECEMBER 31, 1970 REV.1          0039
  TABSCH        DECEMBER 31, 1970 REV.1          0039
  TABSTR        DECEMBER 31, 1970 REV.1          0039
  LSTOUT        DECEMBER 31, 1970 REV.1          0039
  LINK1         DECEMBER 31, 1970 REV.1          0039
  LINK2         DECEMBER 31, 1970 REV.1          0039
  COREXT        DECEMBER 31, 1970 REV.1          0039
  DPRADD        DECEMBER 31, 1970 REV.1          0039
  LOADER        DECEMBER 31, 1970 REV.1          0039
  NAMPRO        DECEMBER 31, 1970 REV.1          0039
  RBDBZS        DECEMBER 31, 1970 REV.1          0039
  ENTEXT        DECEMBER 31, 1970 REV.1          0039
  XFRPRO        DECEMBER 31, 1970 REV.1          0039
  HEXPRO        DECEMBER 31, 1970 REV.1          0039
  EOLPRO        DECEMBER 31, 1970 REV.1          0039
  ADRPRO        DECEMBER 31, 1970 REV.1          0039
*L     ALCORE
  ALCORE        DECEMBER 31, 1970 REV.1          0C77
  ALVOL         DECEMBER 31, 1970 REV.1          0D21
  OFVOL         DECEMBER 31, 1970 REV.1          0D3E
  PARAME        DECEMBER 31, 1970 REV.1          0D4A
  COMMON        DECEMBER 31, 1970 REV.1          0DA8
  NEPROC        DECEMBER 31, 1970 REV.1          0DBF
  NMONI         DECEMBER 31, 1970 REV.1          0E23
  RW            DECEMBER 31, 1970 REV.1          0E65
  MAKQ          DECEMBER 31, 1970 REV.1          0F22
  MINT          DECEMBER 31, 1970 REV.1          0F49
*M        JOBENT
  JOBENT        DECEMBER 31, 1970 REV.1          005A
  T11           DECEMBER 31, 1970 REV.1          005A
  T7            DECEMBER 31, 1970 REV.1          005A
  T5            DECEMBER 31, 1970 REV.1          005A
  T3            DECEMBER 31, 1970 REV.1          005A
*M        JOBPRO
  JOBPRO        DECEMBER 31, 1970 REV.1          005F
*M        PROTEC
  PROTEC        DECEMBER 31, 1970 REV.1          0064
  JBKILL        DECEMBER 31, 1970 REV.1          0064
*M        JPLOAD
  JPLOAD        DECEMBER 31, 1970 REV.1          006F
*M        JPCHGE
  JPCHGE        DECEMBER 31, 1970 REV.1          0074
  ASCHEX        DECEMBER 31, 1970 REV.1          0074
*M        JPT13
  T13           DECEMBER 31, 1970 REV.1          0078
*M        MIPRO
  MIPRO         DECEMBER 31, 1970 REV.1          007D
```

```
*M          RESTOR
  RESTOR          DECEMBER 31, 1970 REV.1          0080
*M          ODEBUG
  ODEBUG          DECEMBER 31, 1970 REV.1          0083
*M          RCOVER
  RCOVER          DECEMBER 31, 1970 REV.1          00A5
  OUTSEL          DECEMBER 31, 1970 REV.1          00A5
  DMPCOR          DECEMBER 31, 1970 REV.1          00A5
  MASDMP          DECEMBER 31, 1970 REV.1          00A5
*M          BRKPT
  BRKPTD          DECEMBER 31, 1970 REV.1          00AE
  BIASCI          DECEMBER 31, 1970 REV.1          00AE
  SIFT            DECEMBER 31, 1970 REV.1          00AE
  RETJMP          DECEMBER 31, 1970 REV.1          00AE
  JUMPTO          DECEMBER 31, 1970 REV.1          00AE
  ENTER           DECEMBER 31, 1970 REV.1          00AE
  ENTCOR          DECEMBER 31, 1970 REV.1          00AE
  PRTREG          DECEMBER 31, 1970 REV.1          00AE
  SETBRP          DECEMBER 31, 1970 REV.1          00AE
  TERMIN          DECEMBER 31, 1970 REV.1          00AE
  DMPCOR          DECEMBER 31, 1970 REV.1          00AE
  MASDMP          DECEMBER 31, 1970 REV.1          00AE
  RESUME          DECEMBER 31, 1970 REV.1          00AE
*M          SELF
  SELFS           DECEMBER 31, 1970 REV.1          00BC
*M          LOGGER
  LOGA            DECEMBER 31, 1970 REV.1          00C1
*M     SYS CHECKOUT
  SYSCOP                                           00C4
*M
  C01ST                                            00C8
  C02ND                                            00C8
  C03RD                                            00C8
  COLAST                                           00C8
*S,FMASK,7FFF
*L     MASDRV
  MASDRV          DECEMBER 31, 1970 REV.1          0FFF
  S13001          DECEMBER 31, 1970 REV.1          1493
*M     MASS MEMORY DRIVERS
  S13002          DECEMBER 31, 1970 REV.1          00FA
*S,M1713R,S
*M
  S13003          DECEMBER 31, 1970 REV.1          00FD
*S,M1713P,S
*M
  PRT40           DECEMBER 31, 1970 REV.1          0100
*S,MAS501,S
*M
  DR1728          DECEMBER 31, 1970 REV.1          0106
*S,MAS28,S
*M
  CD1729          DECEMBER 31, 1970 REV.1          0110
*S,MAS292,S
*M
  CR405           DECEMBER 31, 1970 REV.1          0115
```

```
*S,MAS405,S
*M
  DR1732          DECEMBER 31, 1970 REV.1          0119
*S,MAS32,S
*M
  PUNCDR          DECEMBER 31, 1970 REV.1          0124
*S,TP1777,S
*M
  PTREAD          DECEMBER 31, 1970 REV.1          0127
*S,TR1777,S
*L
  TAPEDR          DECEMBER 31, 1970 REV.1          1612
  FRWA            DECEMBER 31, 1970 REV.1          1731
  FRWB            DECEMBER 31, 1970 REV.1          1806
  RWBA            DECEMBER 31, 1970 REV.1          1909
  RECOVT          DECEMBER 31, 1970 REV.1          198F
  TAPE            DECEMBER 31, 1970 REV.1          1A1E
  DISKWD          DECEMBER 31, 1970 REV.1          1A2C
  SPACE           DECEMBER 31, 1970 REV.1          1BE9
*S,MAS31,7FFF
*S,TIVALU,7FFF
*S,ABUFAL,7FFF
*S,SNAPE,7FFF
*S,PARITY,7FFF
*S,IPROC1,7FFF
*S,T30,7FFF
*S,T29,7FFF
*S,T28,7FFF
*S,T27,7FFF
*S,T26,7FFF
*S,T25,7FFF
*S,T24,7FFF
*S,T23,7FFF
*S,T22,7FFF
*S,T21,7FFF
*S,T20,7FFF
*S,T19,7FFF
*S,T18,7FFF
*S,T17,7FFF
*S,T16,7FFF
*S,T13,7FFF
*S,T11,7FFF
*S,T8,7FFF
*S,T7,7FFF
*S,T5,7FFF
*S,T3,7FFF
*S,DEBUG,7FFF
*S,TIMACK,7FFF
*F
VRFCTN
*T
                                                 016F
```

```
IN

*S,1,0,M
IN

*S,2,3,M
IN

*S,3,0,M
IN

*S,4,1,M
IN

*S,5,2,M
IN

*S,6,3,M
IN

*S,7,2,M
IN

*S,8,2,M
IN

*S,9,2,M
IN

*S,10,2,M
IN

*S,11,2,M
IN

*S,12,3,M
IN

*S,13,3,M
IN

*S,14,3,M
IN

*S,15,3,M
IN

*S,16,3,M
IN

*U
```

6.3.2 LOADING AFTER INITIALIZATION

The mass memory modules may be updated after initialization by using the *M instruction in LIBEDT. To do so, perform the following instructions during initialization.

Assign ordinals (step 1).

Set COBOPS and COBOPL entry points (step 3).

Load COBOP, since it is core resident (step 4).

Noting that both SYSCOP and COLAST must be followed by an *T if the input device is magnetic tape, continue after initialization with the following procedures.

1. Type *LIBEDT

   Press RETURN

   Message LIB
   IN

2. Type *K,Ilu

   lu contains the relocatable binaries of the mass memory modules

   Press RETURN

   Message IN

3. Type *M,19,,M,N

   Press RETURN

   SYSCOP is loaded

   Message IN

4. Type *M,20,,M,N

   Press RETURN

   CO1ST, CO2ND, CO3RD, and COLAST are loaded

   Message IN

6.4 ADDITIONAL PROCEDURES

6.4.1 USER INSTRUCTIONS

Be sure that COBOP is intact before using the following procedures to check for system status or for system malfunctioning.

1. Set the RUN/STEP switch to RUN
2. Clear all registers except the A and Q registers
3. Set the P register to the starting address of COBOP
4. Set the SELECTIVE STOP switch
5. Set the RUN/STEP switch to RUN
6. When the machine stops, select the Q register to examine core data as explained in part III, section 5.1.

   If FFFF appears on the push button register, the hardware malfunctioned. Repeat steps 1 through 6.

   If 0000 appears on the push button register, restart the system.

7. To schedule SYSCOP:

   Press the mannual interrupt button on the teletypewriter

   Message MI

   Type SYSCOP

   Press RETURN

8. Message SELECT OPTION
9. SYSCOP may be rerun as often as necessary on the same image if the area on which the failed image is written is not destroyed. Respond with one of the following five options which will remain in effect until the DUMP routine is executed:

Option	Action	Significance
1	Type *Z Press RETURN Message FINISH SYSCOP	To release package; no further input is necessary but the package may be rescheduled.

Option	Action	Significance
2	Type 0	To transfer control to DUMP routine
	Press RETURN	
	Message DUMP	
	Type one of the following	
	*Z	To exit from the package
	*R	To repeat the SYSCOP package beginning with step 8
	*Dxxxx,yyyy	To dump cells xxxx to cells yyyy from the failed image; DUMP appears as a message to request the next input or if invalid data is typed
	Press RETURN	Return to step 8
3	Type 1	To print error messages only on list logical unit; return to step 8
	Press RETURN	
4	Type 2	To print errors plus supporting messages on list logical unit; return to step 8
	Press RETURN	
5	Type 3	To print errors and all supporting messages on list logical unit; return to step 8
	Press RETURN	

Loading Example

The failed image has already been written on mass memory by COBOP for the following verification of correct loading after the scheduling of SYSCOP:

Typeout/Printout	Significance
SYSCOP START	Output on list lu; indicates SYSCOP is scheduled and has begun
IMAGE START SECTOR IS 2000	
SELECT OPTION	Appears on comment logical unit
0	User types to transfer control to DUMP routine
DUMP	Appears on comment logical unit
*R	User types to repeat the SYSCOP package
SELECT OPTION	Appears on comment logical unit
1	User types to request printing of error messages
. . . errors, if any	Errors appear on list logical unit

Typeout/Printout	Significance
SELECT OPTION	Appears on comment logical unit
*Z	User types to release package
FINISH SYSCOP	Appears on list logical unit; package is complete; core is released

Printout Examples

Option 1:

```
***SYSTEM DIRECTORY ERROR
INDEX 000F HAS INVLAID REQ PRI 0004
INDEX 0001 TOO LONG FOR REQ PRI 0000
INDEX 0014 TOO LONG FOR REQ PRI 0004
```

Option 2:

```
  A       Q       I     REGISTER
0000    0000    1FA2
PRI LVL WAS 0000
LU  04  CURRENT PARA LIST AT 21F5
RC  0800
C   0000
TH  FFFF
LU  1004
N   0010
S   208B
I/O IN PROGRESS
RETURN FOR FNR WAS  14E3
RETURN FOR CMR WAS  1555
LAST ENTRY TO BE SCHEDULED
0360/  12AA   0D25   0364   0091
THERE WERE  0000 OF THE 0101 VOLATILE WORDS ASSIGNED
ALLOCATABLE CORE MAP
```

```
INDEX START LNGTH THRD DUMP
0002   1E56   0144   1E58   C8FE   6C22   40FF   0822   0927
000A   1F9A   041C   1F9C   C8FE   6400   0ABB   0B00   5800
EMPY   23B6   064A   FFFF   C8FE   6400   0ABC   0814   B032
***SYSTEM DIRECTORY ERROR
INDEX 000F HAS INVALID REQ PRI   0004
INDEX 0001 TOO LONG FOR REQ PRI 0000
INDEX 0014 TOO LONG FOR REQ PRI 0004
SYSTEM NOT SWAPPED
JP WAS IN CORE
FILE1   FILE 2  FILE3   FILE4   LOADR BP
1E58    1F9C    0000    0000    0000    0000
JP LOCKED OUT FOR LIBEDT OR RECOVERY
```

Option 3:

```
 A       Q       I    REGISTER
0000    0000    1FA2
MAX CORE WAS 6FFF WITH 2A00 TO 6FFE UNPROT
MAXSEC WAS    00003E7F
MAX CORE WAS  6FFF WITH 2A00 TO 6FFE UNPROT
MAXSEC WAS    00003E7F
PRI  LVL  WAS  0000
LINE   01 LAST INTERRUPTED 21FD
LINE   04 LAST INTERRUPTED 0422
LINE   05 LAST INTERRUPTED 2220
LINE   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
LEVEL F  A  D  B  9  A  6  6  9  9  D  D  6  6  6  6
LINE      1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
LEVEL F  A  D  B  9  A  6  6  9  9  D  D  6  6  6  6
INTRPT STACK LEVEL
 -1
LU   04 CURRENT PARA LIST AT 21F5
RC   0800
C    0000
TH   FFFF
LU   1004
N    0010
S    208B
I/O  IN PROGRESS
RETURN FOR FNR WAS 14E3
RETURN FOR CMR WAS 1555
NUM OF SCHEDL STACK ENTRIES WAS 18
NUM OF SCHEDL CALLS STACKED WAS 00
LAST ENTRY TO BE SCHEDULED
0360/    12AA    0D25    0364    0091
THERE WERE 0000 OF THE 0101 VOLATILE WORDS ASSIGNED
ALLOCATABLE CORE MAP
INDEX START LNGTH THRD DUMP
0002   1E56   0144   1E58   C8FE   6C22   40FF   0822   0927
000A   1F9A   041C   1F9C   C8FE   6400   0ABB   0B00   5800
EMPY   23B6   064A   FFFF   C8FE   6400   0ABC   0814   B032
```

SYSTEM CONFIGURATOR 1.0 7

7.1 RELEASE DESCRIPTION

7.1.1 NEW FEATURES

- System definitions and skeletons have been modified and are available with the MSOS 3.0 release. This update makes it possible to configure all software available with MSOS 3.0.
- New device capabilities are:

  1777 reader/punch

  1751 drum

  1731 buffered magnetic tape

  1726-405 buffered card reader

  1572/1573 timer
- All drivers have mass memory capabilities.
- Encode/decode and re-entrant and non-re-entrant FORTRAN are available for configuration.
- Logical units no longer need to be in interrupt line order.
- MASDRV and DBLDRV are customized according to the user's specifications.

7.1.2 CORRECTIONS

None

7.1.3 KNOWN LIMITATIONS

1. Logical unit numbers input on control statements must be in decimal and must be valid unit numbers for the system. SYSCON attempts to use any decimal number in the range of 1-127; thus, invalid unit numbers will cause SYSCON to terminate with a system J02 error.
2. The restriction described in the first item above is also true for INPUT FROM LOGICAL UNIT parameter phrases associated with the INSERT components.

3. A comma must follow the component name even though no parameters are specified.

4. The System Configurator requires a special version of SPACE which is released on the System Configurator COSY source tape. As released, the system definitions and skeletons are correct. However, if any further modifications to the configurator are necessary, the following code must be added to the System Configurator COSY source tape.

```
SPACE   DCK/ I=6,H=7
        DEL/ 1
        NAM  SPACE          CUSTOMIZED FOR MSOS 3.0
        INS/ 13
        EXT  SYSLVL
        DEL/ 91
        LDA  ALCLGH,Q
        DEL/ 152
        SAZ  SETDIR
        INS/ 156
        JMP* SETDIR
*****************************************************************************
*****************************************************************************
*****************************************************************************
        EXT  JOBENT
        EXT  LIBEDT
        EXT  PROTEC
        EQU  FOUR($25)
        EQU  SYDIR($EB)
*
        SPC  3
        SPC  5
*       THIS SEQUENCE OF CODE SETS THE PRIORITY FOR JOBENT
*       TO FORCE IT INTO THE CORRECT AREA OF ALLOCATABLE CORE
*       IT ALSO SETS THE LIMITS OF SYSTEM DIRECTORY READS FOR
*       THE PROTECT PROCESSOR AND LIBEDT
        SPC  5
*****************************************************************************************
*
        SPC  3
SDJOB   ADC  JOBENT
SDLIB   ADC  LIBEDT
SDPRO   ADC  PROTEC
        SPC  3
*
*****************************************************************************
*
```

```
        SPC  5
SETDIR LDQ- SYDIR
        ADQ* SDJOB
        ENA  $10     SET PRIORITY
        STA- (ZERO),Q
        LDQ- SYDIR
        ADQ* SDLIB   SET LIMITS FOR INITIAL LOAD
        LDA  =N$441 OF LIBEDT TO THE REQUIER LENGTH.
        STA- (FOUR),Q    RELEASE VALUE = $441
        LDQ- SYDIR
        ADQ* SDPRO   SET LIMITS FOR INITIAL LOAD
        LDA  =N$310          OF PROTEC PROCESSOR TO REQUIRED LENGTH
        STA- (FOUR),Q   RELEASE VALUE = $300
        SPC  3
*
*******************************************************************************
*******************************************************************************
        SPC  5
        DEL/ 161
        AND  TIQACK
        DEL/ 165
        LDQ  TIQACK




        INQ  -1              ENABLE INTERRUPT
        DEL/ 169,174
T1572   LDA* ST1572
        LDQ  TIQACK
        OUT  TMRRJT-*        START 1572 TIMER
        INQ  -1
        LDA  =XTIVALU        SET COUNTS VALUE
        DEL/ 190,191
        NUM  $C
        ADC  SUMLVL
        INS/ 245
TIQACK  ADC  TIMACK
        INS/ 250
SUMLVL  NUM  $0D0A
        ALF  7,SUMMARY LEVEL
        ADC  SYSLVL
        NUM  $0D0A
        DEL/ 261,263
        EQU  M1($1AF)        AREA 1
        EQU  M2($1B0)        AREA 2
        EQU  M3($310)        AREA 3
  80  RECORDS TRANSFERRED
```

5. The system definitions and skeletons as released with MSOS 3.0 contains the engineering file set up for the following logical unit order:

lu	Device
1	Core allocator
2	1777 paper tape reader
3	1777 paper tape punch
4	1713 teletypewriter keyboard
5	Dummy
6	1731-601 magnetic tape - unit 0
7	1731-601 magnetic tape - unit 1
8	1738-853 disk - unit 0
9	1740-501 printer
10	1728-430 card reader
11	1729-2 card reader
12	501 FORTRAN line printer
13	1726-405 card reader
14	1732-608 magnetic tape - unit 2
15	1732-608 magnetic tape - unit 3
16	1732-609 magnetic tape - unit 4
17	1732-609 magnetic tape - unit 5
18	1713 reader
19	1713 punch

If the engineering file needs to be changed follow the procedures in part II, section 1.4: then replace the module in the system by using the *M statement in LIBEDT.

7.1.4 KNOWN DEFICIENCIES

None

7.2 REQUIREMENTS

7.2.1 RELEASE MATERIALS

Magnetic Tape Version

One installation tape

One tape containing system definitions and skeletons

Paper Tape Version

Twelve binary installation tapes

Fourteen paper tapes containing system definitions and skeletons

Optional Tapes

One COSY source magnetic tape

One list magnetic tape

7.2.2 HARDWARE REQUIREMENTS

The minimum hardware configuration is the same as for MSOS 3.0. For optimum installation and execution add the following hardware:

Either 2 1731-601 magnetic tape units or
2 1732-608 magnetic tape units

Either 1 1726-405 card reader or
1 1728-430 card reader or
1 1729-2 card reader and 1 1742-501 line printer

7.2.3 MEMORY REQUIREMENTS

A minimum of 3000 words of unprotected core is necessary to execute SYSCON.

7.3 INSTALLATION PROCEDURES

For installation of SYSCON:

mass storage device is LUN 8

paper tape reader is LUN 2

magnetic tape driver (if present) is LUN 6

All other logical units are dependent upon installation.

MSOS 3.0 must be installed

1. Type *LIBEDT

   Press RETURN

   Message LIB
           IN

2. If using magnetic tape,

   a. Mount the installation magnetic tape on LU6, the magnetic tape device

   b. When READY,

      Type *K,I6

      Press RETURN

      Message IN

      Type *V,6

      Press RETURN

      Message IN

   If using paper tape,

   a. Mount the first paper installation tape on LU2, the paper tape reader

   b. Press CLEAR

      Type *K,I2

      Press RETURN

      Message IN

      Type *V,2

      Press RETURN

      Message IN

3. As the tape is used, the System Configurator is installed on the program library, and the following appears on the standard print device:

```
IN
*L, CONFIG
IN
*K, I6, P8
IN
*P, F, GOCONF
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1A      371B
     OPTCHK    372E
     INPREC    379A
     MESSGS    38FA
     SCNOPT    3998
     INITAL    3A06
     CONVRT    3AFD
     CONTRL    3B76
     CORECT    3B98
     INSINP    3C06
     SCNREC    3C1C
IN
*K, I8
IN
*N, CONF1A, , , B
IN
*K, I6
IN
*P, F, GO1B
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1B      371B
     DEFINE    3722
     PARCHK    38E7
     PAMCHK    3916
     PARTIT    3943
```

```
    SEARCH   3963
    SCNREC   39BA
    INPREC   3A4D
    CONTRL   3BAD
    VALPRO   3BCF
    VALCHK   3C1D
    PICKUP   3D3C
IN

*K,I8
IN

*N,CONF1B,,,B
IN

*K,I6
IN

*P,F,GO1C
    CONFIG   3489
    GOCONF   349B
    SCDKIO   349D
    ERROR    34BF
    DCTOAS   351A
    GETITM   3561
    CALADR   365A
    INCPTR   3698
    GETFLE   36B9
    GO1C     371B
    SYSDAT   371F
    SYSINS   377C
    INSINP   37D6
    INCINS   37EC
    GETCHR   37FF
    STOCHR   381D
    WRTMMR   3845
    RDSKEL   387E
    INITCM   38A8
    COMMNT   38E3
IN

*K,I8
IN

*N,CONF1C,,,B
IN

*K,I6
IN
```

```
*P,F,GO1D
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1D      371B
     SPECF1    371F
     PARCHK    38E2
     BKCMVR    3911
     SPCPAR    3970
     SEARCH    39A3
     SCNREC    39FA
     CONTRL    3A8D
     INPREC    3AAF
     CORECK    3C0F
     CORECT    3C2B
     CONVRT    3099
     INSINP    3D12
IN

*K,I8
IN

*N,CONF1D,,,B
IN

*K,I6
IN

*P,F,GO1E
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1E      371B
     SPECF2    371F
     PAMCH2    3823
     INSURT    38DA
     INCINS    397F
     INSINP    3992
     SEARCH    39A8
```

```
     CORECT   39FF
     SCNREC   3A6D
     CONTRL   3B0C
     INPREC   3B22
     CNVTNO   3C82
     PICKUP   3CD8
IN

*K,I8
IN

*N,CONF1E,,,B
IN

*K,I6
IN

*P,F,GO1F
     CONFIG   3489
     GOCONF   3498
     SCDKIO   349D
     ERROR    34BF
     DCTOAS   351A
     GETITM   3561
     CALADR   365A
     INCPTR   3698
     GETFLE   36B9
     GO1F     371B
     VARPRO   3725
     RNGCHK   3823
     SCNREC   390E
     CORECT   39A1
     VALCHK   3A0F
     INPREC   3B2E
     CONTRL   3C8E
     CNVTNO   3CB0
     PICKUP   3D06
IN

*K,I8
IN

*N,CONF1F,,,B
IN

*K,I6
IN

*P,F,GO2
     CONFIG   3489
     GOCONF   349B
     SCDKIO   349D
     ERROR    34BF
     DCTOAS   351A
     GETITM   3561
```

```
      CALADR   365A
      INCPTR   3698
      GETFLE   36B9
      GO2      371B
      PHASE2   371F
      EQUIVA   379C
      INSERT   37D4
      DELETE   38AB
      GETVAL   38E1
      CVTNUM   3985
      GETNUM   39D8
      REDREC   39FA
      GETCHR   3A4F
      GNSCHR   3A6D
      STOCHB   3A7A
      DECASC   3AA2
      MMREAD   3B0D
      OUTREC   3B2E
      HICORE   3B3E
      INTREG   3B73
      PICKUP   3D3D
      P2NAM1   3D4D
IN

*K,I8
IN

*N,CONF2A,,,B
IN

*K,I6
IN

*P,F,GO2
      CONFIG   3489
      GOCONF   349B
      SCDKIO   349D
      ERROR    34BF
      DCTOAS   351A
      GETITM   3561
      CALADR   365A
      INCPTR   3698
      GETFLE   36B9
      GO2      371B
      PHASE2   371F
      EQUIVA   379C
      DELETE   37D4
      GETVAL   380A
      CVTNUM   38AE
      GETNUM   3901
      REDREC   3923
      GETCHR   3978
      GNSCHR   3996
      STOCHR   39A3
```

```
      DECASC   39CB
      MMREAD   3A36
      OUTREC   3A57
      MSKTBL   3A67
      FTNLVL   3B65
      SCHSTK   3BEE
      PICKUP   3C2B
      P2NAM2   3C3B
IN

*K,I8
IN

*N,CONF2B,,,B
IN

*K,I6
IN

*P,F,GO2
      CONFIG   3489
      GOCONF   349B
      SCDKIO   349D
      ERROR    34BF
      DCTOAS   351A
      GETITM   3561
      CALADR   365A
      INCPTR   3698
      GETFLE   36B9
      GO2      371B
      PHASE2   371F
      EQUIVA   379C
      DELETE   37D4
      GETVAL   380A
      CVTNUM   38AE
      GETNUM   3901
      REDREC   3923
      GETCHR   3978
      GNSCHR   3996
      STOCHR   39A3
      DECASC   39CB
      MMREAD   3A36
      OUTREC   3A57
      LUTBLS   3A67
      DGNTAB   3D06
      PICKUP   3D48
      P2NAM3   3D58
IN

*K,I8
IN

*N,CONF2C,,,B
IN
```

```
*K,I6
IN

*P,F,GO2
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO2       371B
     PHASE2    371F
     EQUIVA    379C
     INSERT    37D4
     DELETE    38AB
     GETVAL    38E1
     CVTNUM    3985
     GETNUM    39D8
     REDREC    39FA
     GETCHR    3A4F
     GNSCHR    3A6D
     STOCHR    3A7A
     DECASC    3AA2
     MMREAD    3B0D
     OUTREC    3B2E
     OUTLNN    3B3E
     FTNMSK    3BE3
     PRESET    3C0A
     PICKUP    3C73
     P2NAM4    3C83
IN

*K,I8
IN

*N,CONF2D,,,B
IN

*K,I6
IN

*P,F,GO3A
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
```

```
      GO3A     371B
      PHASE3   371F
      PACKAG   378C
      INSPGM   37F1
      DELPGM   38CA
      OUTCRD   3905
      XTCORE   397E
      INPBIN   3992
      OUTBIN   39E6
      UNLOAD   3A43
      GETVAL   3A84
      CVTNUM   3B28
      GETNUM   3B7B
      GETCHR   3B9D
      GNSCHR   3BBB
      STOCHR   3BC8
      BINASC   3BFC
      PICKUP   3C3D
      PAGEJT   3C4D
      PRNTLN   3C5A
      PACKLN   3C7B
      NEWHDR   3CAF
      STAPGM   3CE2
      STAPCK   3D52
IN

*K,I8
IN

*N,CONF3A,,,B
IN

*K,I6
IN

*P,F,GO3B
      CONFIG   3489
      GOCONF   349B
      SCDKIO   349D
      ERROR    34BF
      DCTCAS   351A
      GETITM   3561
      CALADR   365A
      INCPTR   3698
      GETFLE   36B9
      GO3B     371B
      INPBIN   3722
      GETVAL   3776
      CVTNUM   381A
      GETNUM   386D
      GETCHR   388F
```

```
        GNSCHR    38AD
        BINASC    38BA
        PICKUP    3907
        PAGEJT    3917
        PRNTLN    3924
        PACKLN    3945
        OUTBIN    3979
        STAEND    39D6
    IN

    *K, I8
    IN

    *N, CONF3B, , , B
    IN

    *U
```

4. Message: IN

   Type: *Z

   Press: RETURN

   Message: J

   System Configurator is installed.

7.4 ADDITIONAL PROCEDURES

7.4.1 VERIFICATION OF INSTALLATION

Description

The verification procedure demonstates that SYSCON is correctly installed. It exercises the STATISTICS and CONVERSE options of SYSCON. The following procedure fully covers an optimum as well as the minimum hardware configuration.

Requirements

MSOS must be installed, and the job processor must be in core.

Procedure

1. If using a minimum system,

   a. Mount the first paper tape containing SYSCON definitions and skeletons on LUN2

   b. When LUN2 is ready, continue with step 2

   If using a non-minimum system,

   a. Mount the magnetic tape containing SYSCON definitions and skeletons on LUN6, the magnetic tape device

   b. When LUN6 is ready, continue with step 2

2. Type *P

   Press RETURN

   Message J

3. Type *CONFIG

   Press RETURN

   Message OPTIONS (STATISTICS, CONFIGURE, CONVERSE)

4. Type ST, CONV

   Press RETURN

   Teletypewriter paper advances

5. If using a minimum system,

   Type *J, 2

   Press RETURN

   If using a non-minimum system,

   Type *J, 6

   Press RETURN

6. SYSCON reads in the SYSCON system definitions and skeletons tape.

7. If using a minimum system, the following message appears:

   Message UNLOAD SYSTEM DEFINITIONS, LOAD SYSTEM SPECIFICATIONS

   Remove the system definitions and skeletons paper tape from the paper tape reader.

8. The verification data set program is on the COSY source tape under the deck name VERIFY. If a COSY source tape is not available, the verification data set program is as follows. Transfer the program to either paper tape, cards, or magnetic tape. Place the verification data set program on the input device to be used.

```
**          V E R I F I C A T I O N   D E C K   F O R   S Y S C O N
**      COPYRIGHT CONTROL DATA CORP.  1970
**          S P E C I F I C A T I O N   L I S T
**
*S Y S T E M   H A R D W A R E   D E V I C E S
**
**      INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
*+1703,
**      1777 READER/PUNCH
*+1777,
**      1711/1712 TELETYPE
*+1711,
**      1738 DISK CONTROLLER WITH 853-4 DISK DRIVES
*+1738/853-4,
**
*C O R E   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
**
**      MSOS*3.0 MONITOR
*+MONITOR,
**
*M A S S   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
**
**      JOB PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
*+JOB PROCESSOR,
**
**
*P R O G R A M   L I B R A R Y   P R O G R A M S
**
*+FTN RUNTIME LIBRARY,
**
*TERMINATE
```

9. Type *I,lun

   lun is the device to be used

   Press RETURN

10. SYSCON reads in the verification data set program. After several records are read in, the following message appears:

    Message ERROR,10,5

11. Type **

    Press RETURN

12. Type *V

    Press RETURN

    SYSCON continues to read in the verification data set program until it is completely in.

    Message J

13. If the output is similar to that given in steps 14 and 16 or 15 and 16, SYSCON is installed correctly.

14. If the list device is other than the teletypewriter, the following is a sample of the printout which appears on the list device.

```
OPTIONS (STATISTICS, CONFIGURE, CONVERSE)
     1   ST,CONV
     2   *J,14
     3   *I,11
         **          V E R I F I C A T I O N   D E C K   F O R   S Y S C O N
         **      COPYRIGHT CONTROL DATA CORP.  1970
         **          S P E C I F I C A T I O N   L I S T
         **
     4   *S Y S T E M   H A R D W A R E   D E V I C E S
         **
         **      INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
     5   *+1703,
ERROR,     10,       5

         **      1777 READER/PUNCH
     6   *V
     7   *+1777,
         **      1711/1712 TELETYPE
     8   *+1711,
         **      1738 DISK CONTROLLER WITH 853-4 DISK DRIVES
     9   *+1738/853-4,
         **
    10   *C O R E   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
         **
         **      MSOS*3.0 MONITOR
```

```
11  *+MONITOR,
    **
12  *M A S S   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
    **
    **      JOB PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
13  *+JOB PROCESSOR,
    **
    **
14  *P R O G R A M   L I B R A R Y   P R O G R A M S
    **
15  *+FTN RUNTIME LIBRARY,
    **
16  *TERMINATE
```

15. If the list device is the teletypewriter, information similar to the following appears on the teletypewriter:

```
*CONFIG
OPTIONS (STATISTICS, CONFIGURE, CONVERSE)

ST, CONV
    1   ST, CONV
*J, 7
    2   *J, 7
*I, 2
    3   *I, 2
        **          VERIFICATION DECK FOR SYSCON
        **          SPECIFICATION LIST
        **
    4   *SYSTEM HARDWARE DEVICES
        **
        **      INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
    5   *+1703,
ERROR,  10,      5

*+1721,
        **      1723/1724 PAPER TAPE PUNCH
    6   *+1721,
*V
    7   *V
```

16. Data similar to the following appears on the list device after the information in steps 14 or 15.

```
                    SYSTEM STATISTICS

              CORE RESIDENT FOREGROUND PROGRAMS


          MONITOR PROGRAMS
TRVEC          44 P          0 D          0 C
DRCORE        320 P          0 D          0 C
NIPROC        126 P          0 D          0 C

 LIBEDT AND ENGINEERING FILE
EF            420 P          0 D          0 C
LIBEDT       4826 P          0 D          0 C


COMMON         23 P          0 D          0 C
NEPROC        100 P          0 D          0 C
NMONI          66 P          0 D          0 C
PARAME         94 P          0 D          0 C
ALVOL          29 P          0 D          0 C
OFVOL          12 P          0 D          0 C
NCMPRQ         49 P          0 D          0 C
NFNR          106 P          0 D          0 C
ALCORE        170 P          0 D          0 C
MINT          182 P          0 D          0 C
RW            189 P          0 D          0 C
MAKQ           39 P          0 D          0 C
ADEV          480 P          0 D          0 C
SCHEDU        170 P          0 D          0 C
NDISP          60 P          0 D          0 C
```

MASS RESIDENT FOREGROUND PROGRAMS

JOB	PROCESSOR			
LOAD	301	P	0 D	0 C
BRANCH	734	P	0 D	0 C
LIDRIV	94	P	0 D	0 C
LCDRIV	45	P	0 D	0 C
LMDRIV	34	P	0 D	0 C
LLDRIV	14	P	0 D	0 C
SCAN	183	P	0 D	0 C
CHPU	11	P	0 D	0 C
ADJOVF	21	P	0 D	0 C
CONVRT	24	P	0 D	0 C
TABSCH	31	P	0 D	0 C
TABSTR	41	P	0 D	0 C
LSTOUT	47	P	0 D	0 C
LINK1	29	P	0 D	0 C
LINK2	43	P	0 D	0 C
COREXT	54	P	0 D	0 C
DPRADD	23	P	0 D	0 C
LOADER	216	P	0 D	0 C
NAMPRO	215	P	0 D	0 C
RBDBZS	259	P	0 D	0 C
ENTEXT	134	P	0 D	0 C
XFRPRO	21	P	0 D	0 C
HEXPRO	266	P	0 D	0 C
EOLPRO	141	P	0 D	0 C
ADRPRO	93	P	0 D	0 C
JOBENT	164	P	0 D	0 C
T11	42	P	0 D	0 C
T7	119	P	0 D	0 C
T5	49	P	0 D	0 C
T3	47	P	0 D	0 C
JOBPRO	422	P	0 D	0 C
PROTEC	945	P	0 D	0 C
JBKILL	104	P	0 D	0 C
JPLOAD	389	P	0 D	0 C
JPCHGE	190	P	0 D	0 C
ASCHEx	107	P	0 D	0 C
T13	422	P	0 D	0 C
RESTOR	216	P	0 D	0 C
RCOVER	337	P	0 D	0 C
OUTSEL	71	P	0 D	0 C
DMPCOR	173	P	0 D	0 C
MASDMP	245	P	0 D	0 C
BRKPTD	198	P	0 D	0 C
BIASCI	90	P	0 D	0 C
SIFT	141	P	0 D	0 C
RETJMP	23	P	0 D	0 C
JUMPTO	22	P	0 D	0 C
ENTER	22	P	0 D	0 C
ENTCOR	33	P	0 D	0 C
PRTREG	53	P	0 D	0 C
SETBRP	117	P	0 D	0 C
TERMIN	73	P	0 D	0 C
DMPCOR	131	P	0 D	0 C
MASDMP	241	P	0 D	0 C
RESUME	195	P	0 D	0 C

```
SELFS        395 P        0 D        0 C
LOGA         222 P        0 D        0 C
```

CORE RESIDENT FOREGROUND PROGRAMS

1713 KEYBOARD

MASS RESIDENT DRIVERS

INPUT/OUTPUT DRIVERS

```
STCK          76 P        0 D        0 C
PTREAD       180 P        0 D        0 C
PUNCDR       173 P        0 D        0 C
TELTYP       316 P        0 D        0 C
DISK         184 P        0 D        0 C
```

SPACE PROGRAM

```
SPACE       1669 P        0 D        0 C
```

PROGRAM LIBRARY PROGRAMS

FORTRAN RUNTIME NON-REENTRANT, NON-RUNANYWHERE LIBRARY

```
Q8EXPN       157 P        0 D        0 C
Q8PRMS        16 P        0 D        0 C
Q8AB          17 P        0 D        0 C
IFALT         22 P        0 D        0 C
SIGN          36 P        0 D        0 C
FXFL          96 P        0 D        0 C
EXPPRG       161 P        0 D        0 C
SQRTF         92 P        0 D        0 C
LNUPRG       116 P        0 D        0 C
TANH         111 P        0 D        0 C
SINCOS       199 P        0 D        0 C
ARCTPG       149 P        0 D        0 C
FLOAT        563 P        0 D        0 C
Q8QINI       202 P        0 D        0 C
Q8QEND        24 P        0 D        0 C
Q8CMP        191 P        0 D        0 C
Q8RWBU       277 P        0 D        0 C
Q8ERRM       211 P        0 D        0 C
Q8DFIO       174 P        0 D        0 C
Q8QX          98 P        0 D        0 C
Q8QUNI        88 P        0 D        0 C
Q8FGET       111 P        0 D        0 C
```

Q8MAGT	75	P	0	D	0	C
TAPCON	170	P	0	D	0	C
IOCK	28	P	0	D	0	C
PSSTOP	55	P	0	D	0	C
Q8PAND	85	P	0	D	0	C
Q8EXP9	173	P	0	D	0	C
Q8EXP1	123	P	0	D	0	C
Q8IFRM	63	P	0	D	0	C
Q8FS	469	P	0	D	0	C
Q8TRAN	1750	P	0	D	0	C
FORTRA	288	P	0	D	0	C
IOCODE	59	P	0	D	0	C
IGETCH	23	P	0	D	0	C
IPACK	58	P	0	D	0	C
UPDATE	10	P	0	D	0	C
DECPL	29	P	0	D	0	C
INTGR	38	P	0	D	0	C
SPACEX	19	P	0	D	0	C
HOLRTH	63	P	0	D	0	C
DCHX	112	P	0	D	0	C
HXASC	52	P	0	D	0	C
AFRMOT	35	P	0	D	0	C
RFRMOT	17	P	0	D	0	C
AFRMIN	44	P	0	D	0	C
RFRMIN	18	P	0	D	0	C
ASCII	20	P	0	D	0	C
HXDC	137	P	0	D	0	C
FLOTIN	64	P	0	D	0	C
FOUT	103	P	0	D	0	C
EOUT	222	P	0	D	0	C
EWRITE	14	P	0	D	0	C
AFORM	27	P	0	D	0	C
RFORM	27	P	0	D	0	C
HEXASC	23	P	0	D	0	C
HEXDEC	29	P	0	D	0	C
DECHEX	31	P	0	D	0	C
FLOATG	28	P	0	D	0	C
FORMTR	420	P	0	D	0	C
INITL1	42	P	0	D	0	C
ASCHX	44	P	0	D	0	C
PSUEDO	37	P	0	D	0	C
Q8QFI	22	P	0	D	0	C
Q8QFL	43	P	0	D	0	C
Q8QFX	48	P	0	D	0	C
Q8QGTX	4	P	0	D	0	C

CORE MEMORY MAP

CORE RESIDENT	1433
ALLOCATABLE	1655
UNPROTECTED	29680
SYSTEM COMMON	0
NON-SYSTEM	0

7.4.2 INSTALLATION OF MSOS 3.0 SYSTEM GENERATED BY CONFIGURATOR

After specifying and configuring a system, use the system initializer to install the new system with the system installation programs and the relocatable binary SYSDAT programs.

Minimum System

If the only input device is a paper tape reader, the installation procedures are basically the same as those described in part I, section 1.3.

1. Load and execute the system initializer as in part I, section 1.3.2, 1.3.3, or 1.3.4.

2. Continue installing using the procedures in part I, section 1.3.5, replacing step 3 of these procedures with:

   Type *S, MAXSEC, xxxx

   Press RETURN

   Message Q

3. When the system initializer types Q in step 7, mark the leader of tape under the paper tape reader and remove the paper tape.

4. Mount the relocatable binary SYSDAT program paper tape in the paper tape reader.

5. Type *V

   Press RETURN

   Message LUN, lun, FAILED
   ACTION

   The paper tape reader is out of tape; mount the paper tape containing the system installation programs placing the leader marked in step 3 under the reader.

6. Type RP

   Press RETURN

7. Continue with the installation as described in step 8 of part I, section 1.3.5.

Non-Minimum System

Use the following instructions for a system containing a paper tape reader and another device.

1. Load and execute the system initializer as described in part I, section 1.3.2, 1.3.3, or 1.3.4.

2. Load input device a with the system installation programs.

3. Load input device b with the relocatable binary of the SYSDAT program.

4. Continue installing using the instructions in part I, section 1.3.5, replacing step 3 of these procedures with:

   Type *S, MAXSEC, xxxx

   Press RETURN

   Message Q

5. Use the system initializer to assign the output and comment devices as discussed in part I, section 1.3.5, step 5.

6. Assign input device a (containing system installation programs) as the input device.

   Type *I, a

   Press RETURN

   Message Q

7. Type *V

   Press RETURN

   Message Q

8. Assign input device b (containing the relocatable binary of the SYSDAT program) as the input device.

   Type *I, b

   Press RETURN

   Message Q

9. Type *

   Press RETURN

10. The relocatable binary of the SYSDAT program is loaded into the system.

    Message ACTION

11. Type CU

    Press RETURN

12. Assign input device a as the input device.

    Type *I,a

    Press RETURN

    Message Q

13. Type *

    Press RETURN

14. Continue the installation with step 8 of part I, section 1.3.5.

Core Map of Configured and Installed System

```
*YM,EFILE,1
*YM,LIBEDT,2
*YM,LOADSD,3
*YM,JOBENT,4
*YM,JOBPRO,5
*YM,PROTEC,6
*YM,JPLOAD,7
*YM,JPCHGE,8
*YM,JPT13,9
*YM,MIPRO,10
*YM,RESTOR,11
*YM,ODEBUG,12
*YM,RCOVER,13
*YM,BRKPT,14
*YM,SELF,15
*YM,LOGGER,16
*          1700 MASS STORAGE OPERATING SYSTEM
*
*          SYSTEM DATA PROGRAM
*L
*U
```

```
  SYSDAT        COPYRIGHT CONTROL DATA CORP. 1970 0000
  DBLDRV        DECEMBER 31, 1970 REV.1           06C3
*V
*         MONITOR PROGRAMS
*L
  TRVEC         DECEMBER 31, 1970 REV.1           0F17
  DRCORE        DECEMBER 31, 1970 REV.1           0F43
  NIPROC        DECEMBER 31, 1970 REV.1           1083
*    LIBEDT AND ENGINEERING FILE
*M        EFILE
  EF            DECEMBER 31, 1970 REV.1           0001
*M        LIBEDT
  LIBEDT        DECEMBER 31, 1970 REV.1           0006
*L
  COMMON        DECEMBER 31, 1970 REV.1           1101
  NEPROC        DECEMBER 31, 1970 REV.1           1118
  NMONI         DECEMBER 31, 1970 REV.1           117C
  PARAME        DECEMBER 31, 1970 REV.1           11BE
  ALVOL         DECEMBER 31, 1970 REV.1           121C
  OFVOL         DECEMBER 31, 1970 REV.1           1239
  NCMPRQ        DECEMBER 31, 1970 REV.1           1245
  NFNR          DECEMBER 31, 1970 REV.1           1276
  ALCORE        DECEMBER 31, 1970 REV.1           12E0
  MINT          DECEMBER 31, 1970 REV.1           138A
  RW            DECEMBER 31, 1970 REV.1           1440
  MAKQ          DECEMBER 31, 1970 REV.1           14FD
  ADEV          DECEMBER 31, 1970 REV.1           1524
  SCHEDU        DECEMBER 31, 1970 REV.1           1704
  NDISP         DECEMBER 31, 1970 REV.1           17AE
  TMINT         DECEMBER 31, 1970 REV. 1          17FA
  DTMER         DECEMBER 31, 1970 REV. 1          187C
  BUFALC        DECEMBER 31, 1970 REV. 1          1899
*         JOB PROCESSOR
*M        LOADSD
  LOAD          DECEMBER 31, 1970 REV.1           0039
  BRANCH        DECEMBER 31, 1970 REV.1           0039
  LIDRIV        DECEMBER 31, 1970 REV.1           0039
  LCDRIV        DECEMBER 31, 1970 REV.1           0039
  LMDRIV        DECEMBER 31, 1970 REV.1           0039
  LLDRIV        DECEMBER 31, 1970 REV.1           0039
  SCAN          DECEMBER 31, 1970 REV.1           0039
  CHPU          DECEMBER 31, 1970 REV.1           0039
  ADJOVF        DECEMBER 31, 1970 REV.1           0039
  CONVRT        DECEMBER 31, 1970 REV.1           0039
  TABSCH        DECEMBER 31, 1970 REV.1           0039
  TABSTR        DECEMBER 31, 1970 REV.1           0039
  LSTOUT        DECEMBER 31, 1970 REV.1           0039
  LINK1         DECEMBER 31, 1970 REV.1           0039
  LINK2         DECEMBER 31, 1970 REV.1           0039
  COREXT        DECEMBER 31, 1970 REV.1           0039
  DPRADD        DECEMBER 31, 1970 REV.1           0039
  LOADER        DECEMBER 31, 1970 REV.1           0039
  NAMPRO        DECEMBER 31, 1970 REV.1           0039
  RBDBZS        DECEMBER 31, 1970 REV.1           0039
  ENTEXT        DECEMBER 31, 1970 REV.1           0039
  XFRPRO        DECEMBER 31, 1970 REV.1           0039
  HEXPRO        DECEMBER 31, 1970 REV.1           0039
  EOLPRO        DECEMBER 31, 1970 REV.1           0039
  ADRPRO        DECEMBER 31, 1970 REV.1           0039
```

```
*M          JOBENT
  JOBENT          DECEMBER 31, 1970 REV.1          005A
. T11             DECEMBER 31, 1970 REV.1          005A
  T7              DECEMBER 31, 1970 REV.1          005A
  T5              DECEMBER 31, 1970 REV.1          005A
  T3              DECEMBER 31, 1970 REV.1          005A
*M          JOBPRO
  JOBPRO          DECEMBER 31, 1970 REV.1          005F
*S,ONE,$7FFF
*S,TWO,$7FFF
*S,THREE,$7FFF
*M          PROTEC
  PROTEC          DECEMBER 31, 1970 REV.1          0064
  JBKILL          DECEMBER 31, 1970 REV.1          0064
*S,WDADR,1
*M          JPLOAD
  JPLOAD          DECEMBER 31, 1970 REV.1          006F
*M          JPCHGE
  JPCHGE          DECEMBER 31, 1970 REV.1          0074
  ASCHEX          DECEMBER 31, 1970 REV.1          0074
*M          JPT13
  T13             DECEMBER 31, 1970 REV.1          0078
*M          MIPRO
  MIPRO           DECEMBER 31, 1970 REV.1          007D
*S,SYSCOP,7FFF
*M          RESTOR
  RESTOR          DECEMBER 31, 1970 REV.1          0080
*M          ODEBUG
  ODEBUG          DECEMBER 31, 1970 REV.1          0083
*M          RCOVER
  RCOVER          DECEMBER 31, 1970 REV.1          00A5
  OUTSEL          DECEMBER 31, 1970 REV.1          00A5
  DMPCOR          DECEMBER 31, 1970 REV.1          00A5
  MASDMP          DECEMBER 31, 1970 REV.1          00A5
*M          BRKPT
  BRKPTD          DECEMBER 31, 1970 REV.1          00AE
  BIASCI          DECEMBER 31, 1970 REV.1          00AE
  SIFT            DECEMBER 31, 1970 REV.1          00AE
  RETJMP          DECEMBER 31, 1970 REV.1          00AE
  JUMPTO          DECEMBER 31, 1970 REV.1          00AE
  ENTER           DECEMBER 31, 1970 REV.1          00AE
  ENTCOR          DECEMBER 31, 1970 REV.1          00AE
  PRTREG          DECEMBER 31, 1970 REV.1          00AE
  SETBRP          DECEMBER 31, 1970 REV.1          00AE
  TERMIN          DECEMBER 31, 1970 REV.1          00AE
  DMPCOR          DECEMBER 31, 1970 REV.1          00AE
  MASDMP          DECEMBER 31, 1970 REV.1          00AE
  RESUME          DECEMBER 31, 1970 REV.1          00AE
*M          SELF
  SELFS           DECEMBER 31, 1970 REV.1          00BC
*M          LOGGER
  LOGA            DECEMBER 31, 1970 REV.1          00C1
*       1713 KEYBOARD
*L
```

```
  S13001          DECEMBER 31, 1970 REV.1            1904
*         MASS RESIDENT DRIVERS
*M
  S13002          DECEMBER 31, 1970 REV.1            00C4
*S,M1713R,S
*M
  S13003          DECEMBER 31, 1970 REV.1            00C7
*S,M1713P,S
*M
  PTREAD          DECEMBER 31, 1970 REV.1            00CA
*S,TR1777,S
*M
  PUNCDR          DECEMBER 31, 1970 REV.1            00CD
*S,TP1777,S
*M
  PRT40           DECEMBER 31, 1970 REV.1            00D0
*S,MAS501,S
*M
  DR1728          DECEMBER 31, 1970 REV.1            00D6
*S,MAS28,S
*M
  CD1729          DECEMBER 31, 1970 REV.1            00E0
*S,MAS292,S
*M
  CR405           DECEMBER 31, 1970 REV.1            00E5
*S,MAS405,S
*M
  DR1732          DECEMBER 31, 1970 REV.1            00EA
*S,MAS32,S
*         INPUT/OUTPUT DRIVERS
*L
  TAPEDR          DECEMBER 31, 1970 REV.1            1A83
  FRWA            DECEMBER 31, 1970 REV.1            1BA2
  FRWB            DECEMBER 31, 1970 REV.1            1C77
  RWBA            DECEMBER 31, 1970 REV.1            1D7A
  RECOVT          DECEMBER 31, 1970 REV.1            1E00
  TAPE            DECEMBER 31, 1970 REV.1            1E8F
*S,MAS31,7FFF
*L
  DISKWD          DECEMBER 31, 1970 REV.1            1E9D
*         SPACE PROGRAM
*L
  SPACE           CUSTOMIZED FOR MSOS 3.0            205A
*F
             SUBPROGRAMS WITH THE FOLLOWING ENTRY POINT
             NAMES HAVE NOT BEEN LOADED DURING *L LOAD.
```

```
   SNAPI
   CM1728
   FF1728
   PARITY
   IPROC1
   SECPRO
   T30
   T29
   T28
   T27
   T26
   T25
   T24
   T23
   T22
   T21
   T20
   T19
   T18
   T17
   T16
   T13
   T11
   T7
   T5
   T3
   DEBUG
   SYSLVL
*T        END
SYSTEM                                    014A
```

CONFIGURATION 1

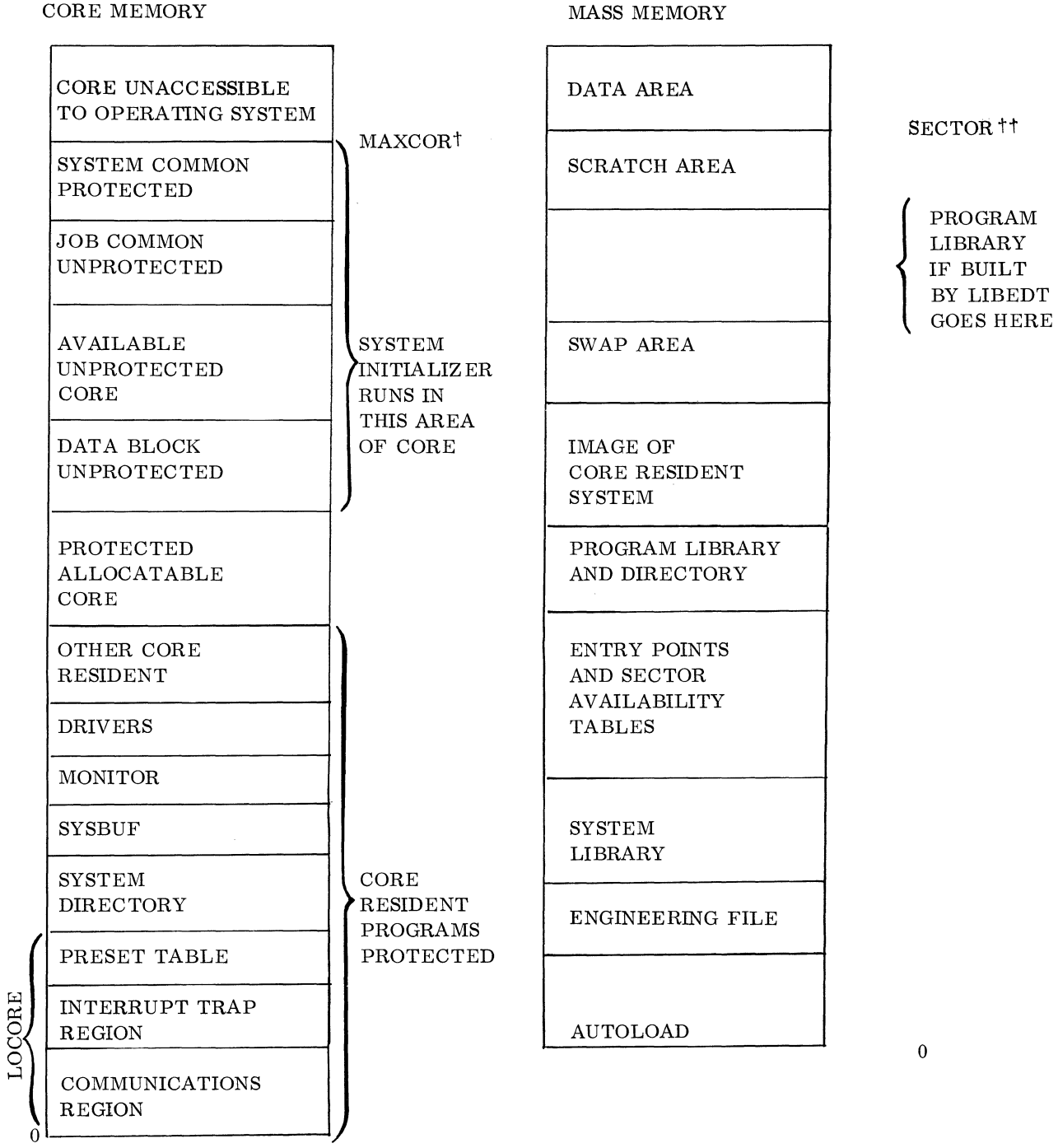
The first section in this part defines and explains the areas which are usually modified to customize MSOS 3.0. Section 1 describes the segments of MSOS as:

Section 2 lists standard drivers and the changes which must be made to the configuration described in section 1 to add these drivers to the MSOS 3.0 system. (Those preceded by an asterisk in the list below are already on the released system.) Special information on the 1706 buffered data channel (section 2.1) and also special instructions for any driver to be installed on mass memory (section 2.15) are also in this section. The outline of section 2 is:

Section 3 defines other possible modifications:

	Section
Building of an initializer	3.1
Manual Input for process program (MIPRO)	3.2
User request modules	3.3
Re-entrant FORTRAN library package	3.4
Non-re-entrant FORTRAN library package	3.5
Output message buffering package	3.6

The diagram below identifies the locations of MSOS 3.0 modules.



† Parameter specified during system initialization determines this area. Section 1.1.1.

†† Parameter specified during system initialization specifies limit of available core.

1.1 LOCORE

The LOCORE program consists of data to be loaded into the communications region, interrupt traps, and preset table. During system initialization, the LOCORE program must be the first *L program loaded after the *Y, *YM system directory entries.

NOTE

If any core-resident system directory entry (*Y) is included, the ordinal must be two or greater, since the first program loaded was LOCORE. LOCORE cannot be a system directory entry.

Part one of the LOCORE program corresponds to the communications region.

Part two is the interrupt trap region from location 100_{16} to the maximum interrupt trap region used (which could be up to a maximum of $13F_{16}$).

Part three of the LOCORE program is the table of presets specifying the name and location of entry points to any protected routines which are also available to unprotected programs.

Part four is designated for use by the assembler or FORTRAN compiler and includes the maximum sector number of the scratch mass storage device.

The following modifications must be made by the system programmer for a specific system.

1.1.1 EQUIVALENCES

MAXCOR is the highest core memory address in hexadecimal available to the system. Core locations above MAXCOR are not affected by normal system operation and may be used for upper core routines, core dumps, etc. This parameter is derived by setting MAXCOR with a *S, MAXCOR, xxxx parameter during system initialization.

1.1.2 COMMUNICATIONS REGION

If required, communications region information can be inserted in the area from location 47_{16} through $B2_{16}$. These entries may be either numeric or the symbolic address of an entry point in another program. In the latter case, the symbolic address must also be declared as an external (EXT). Labels can be attached to these entries and, if declared as entry points (ENT), they can be referenced by other programs. Unused entries should be set to zero.

In the example below, the sequence of code replaces the block:

```
BZS ($B2-$47+1)
```

When the program with entry points SNAPE and SNAPI is loaded, the initializer loads the addresses of SNAPE and SNAPI. Also, it stores a special table starting at location 52_{16} which may be referenced by the entry point name MTAB.

Label	Op	Address
	BZS	($50-$47+1)
	ADC	SNAPE
	ADC	SNAPI
MTAB	NUM	$F
	NUM	$F0
	NUM	$F00
	NUM	$F000
	BZS	(B2-*+1)
	EXT	SNAPE, SNAPI
	EXT	MTAB

1.1.3 INTERRUPT TRAP REGION

The interrupt trap region extends from location 100_{16} to $13F_{16}$ in LOCORE. A four-location trap is necessary for each of the 16 interrupt lines which are used (Section 1.2.3, Interrupt Mask Table). For example, the LINE0 trap area contains four words beginning at word 100_{16}. LINE1 trap area then begins at word 104_{16}. The form of the four-word trap is:

Word	Label	Op	Address
1	LINEno.	NUM	0
2		RTJ-	($FE)
3		NUM	level
4		ADC	interrupt response routine

Explanation of Each Location

Word 1

The hardware stores the state of overflow indicator in bit 15 and also stores the P register contents in bits 14-00. The P register contains the address of the next instruction to be executed when the program is later re-entered.

Word 2

The second word in the interrupt trap is normally used to pass control to the common interrupt handler which will:

1. Store the contents of the A, Q, P, and I registers and the current priority level (PRVL).
2. Establish priority of the program being entered through the third word of the trap.
3. Jump to the interrupt response routine through the fourth word of the trap.

Usually all interrupt lines (except for line 0) use the common interrupt handler whose address is in location FE_{16}.

Any special interrupt handler routines may be used to avoid the overhead required to go through the common interrupt handler. Include the address of the special interrupt handler routines in the communications region between locations 47_{16} and $B2_{16}$ and declare this address as external. The special interrupt handler must preserve the A, Q, P, and I registers and the overflow indicator and return control (with interrupts enabled) to the interrupted program after processing the interrupt. Save priority levels (PRVL) if the response routine runs with interrupts enabled.

Word 3

In word 3 is the priority level of the program which will process the interrupts on the specified line. When assigning priorities:

1. The number in word 3 must correspond with the interrupt mask table entry in MASKT of the SYSBUF or the TABLES program.
2. Priority levels assigned to peripheral devices cannot also be assigned to FORTRAN programs.
3. Because of timing problems, use caution when assigning priorities to devices which are subject to losing data. High priorities should be assigned to these devices, such as the 1729-2 card reader and magnetic tape devices.
4. Interrupt lines for I/O drivers must be assigned the same priority level as that specified in the PHYSTB. That is, the initiator (CP in the appropriate PHYSTB) and the continuator (priority level PR in the appropriate interrupt trap entry) must be the same priority level.

Word 4

This is the address of the interrupt response routine which is the program which processes the interrupt. Each interrupt response routine name must be declared as an external in LOCORE.

External Interrupt Processor (EPROC)

EPROC is a generalized External Interrupt Processor. To use EPROC:

1. Declare it as external in LOCORE.
2. Device must return bit 2 as interrupt status upon a status request.
3. Add the SECPRO table to the SYSBUF program.

SECPRO is a 16-word table which is required only if EPROC is in use. It contains one word for each interrupt line. When EPROC cannot determine which device on a particular line caused an interrupt (indicated by bit 2 of device status), EPROC transfers control to the corresponding secondary processor for that line. SECPRO may contain up to 16 secondary processor addresses. Each location may refer to an entry point of a secondary interrupt processor. The first location of the table is declared entry point SECPRO. (Part I, section 1.2.2, LOG1A Table and EPROC.) Limitations for using EPROC are as follows.

Using EPROC instead of separate response routines for each line increases the interrupt processing time.

Using any of the following special devices requires separate interrupt response routines:

1572 programmable sample rate unit

1573 line synchronized timing generator

Devices which do not give the interrupt status in bit 2 of the A register while a reply is being made to a status command.

Logical units must be specified in line order.

Individual Interrupt Response Routines

Use the following rules when developing individual interrupt response routines.

1. If several devices are driven on the same interrupt line, the interrupt response routine must examine the status of each device to determine which one interrupted.
2. All interrupt response routines for drivers must branch to the driver's continuator entry with the address of the PHYSTB of the interrupting logical unit in the Q register.
3. Interrupt response routines usually reside in the SYSBUF program, except for EPROC.
4. Declare the address of each interrupt response routine as an external in LOCORE.

1.1.4 TABLE OF PRESET ENTRY POINTS

Definition

The preset table is a list of entry points of all programs in protected core, as well as all core-resident subprograms which can be used by jobs running in unprotected core.

Format

This is an example of a preset table entry. If the name of the entry point to the routine is NAME, the following code is required to add NAME to the preset table. The first entry must be for JPRETN.

Label	Op	Address	Comments
	ALF	3, NAME	
ALF	ADC	NAME	
	EXT	NAME	

The EQU for the table length must follow the last entry.

	EQU	LPRSET	(*APRSET) FOR THE LAST ENTRY

Rules

Use caution in constructing the preset table.

The preset table must contain only references to subprograms which cannot destroy the integrity of the protected system.

Subprograms which are referenced in the preset table must be re-entrant if they are also to be used by protected programs. They must have an IIN instruction immediately following each entry point. However, they do not need to be re-entrant if they are not to be used by protected programs.

Location

The preset table begins immediately following the interrupt trap region. The table starts at location 140_{16} if 16 interrupt lines are assigned. The table length is saved at location $F1_{16}$. The table starting address is saved at location $F2_{16}$.

1.1.5 MAXIMUM SCRATCH SECTOR NUMBER (MAXSEC)

Following the preset table is an area reserved for the use of the compiler or the assembler. The maximum sector number available on the scratch mass memory device (MAXSEC) is included in this area. MAXSEC is an initialization time parameter. This parameter is derived by setting SECTOR with an *S, SECTOR, xxxx parameter during system initialization. If part of mass storage is to be reserved for data storage not available to the system, MAXSEC is set to the maximum minus the amount reserved for data.

The area is defined as follows.

Label	Op	Address	Comments
	ENT	MAXSEC	
	BZS	(3)	
	NUM	0	MSB OF MAX SECTOR

Label	OP	Address	Comments
MAXSEC	NUM	SECTOR	LSB of MAC SECTOR
	BZS	(2)	

1.2 SYSBUF

The system and buffer tables program includes the following:

	Section
For use by the operating monitor:	
Logical unit tables	1.2.2
Volatile storage for re-entrant routines	1.2.4
Interrupt and scheduler stacks	1.2.5, 1.2.6
Diagnostic timer table	1.2.9
Routines and tables required by drivers:	
Physical device tables	1.2.12
Interrupt response routines	1.2.13
Output message buffering package	3.7
Special routines:	
Dummy driver and device table	1.2.8
Idle loop routine	1.2.8
Overlay subroutine	1.2.11

1.2.1 EQUIVALENCES (EQU)

Set up SYSBUF equivalences (EQU) as required. Following is a list of EQU's.

EQU	Significance
NUMPRI	Defines the total number of priority levels used by the operating system
NINTLV	Defines the number of priority levels used by interrupts
NFTNLV	Defines the number of priority levels using the re-entrant FORTRAN library
NEDLVL	Defines the number of priority levels using the re-entrant encode/decode package

EQU	Significance
NSR	Defines the maximum number of programs that the timer program may schedule when a single timer interrupt occurs. Delete if the timer is not used
TIMACK	Defines the 1572/1573 timer interrupt acknowledge code. Delete if the timer is not used
TIMCPS	Defines the 1573 timer frequency (Hz). Delete if timer is not used
TODLVL	Defines time-of-day routine request code and priority level. Delete if no time-of-day routine is used

Equivalences are included at appropriate locations in the LOG1A table to identify system logical units.

EQU	Significance
STDINP	Logical unit number of the standard input device, e.g., paper tape or card reader or magnetic tape
BINOUT	Logical unit number of the standard binary output device, e.g., paper tape or card punch or magnetic tape
LSTOUT	Logical unit number of the standard print output device, e.g., teletypewriter or line printer
INPCOM	Logical unit number of the standard input comment device, e.g., teletypewriter
OUTCOM	Logical unit number of the standard output comment device, e.g., teletypewriter
LBUNIT	Logical unit number of the library mass storage device, e.g., disk or drum
SCRTCH	Logical unit number of the scratch mass storage device, e.g., disk or drum
DUMALT	Logical unit number of the dummy device driver

1.2.2 LOGICAL UNIT TABLES

The logical unit tables contain information for all logical units.

LOG1A contains the addresses of physical equipment tables for each logical unit. The order of these addresses reflects the logical assignment of the physical devices in LOG1A.

LOG1 contains the operational flags and alternate logical unit assignments.

LOG2 contains the top of request thread for each logical unit.

Each logical unit number has a corresponding entry in these tables. When using EPROC, the logical units are grouped according to which interrupt line they use. For example, devices which interrupt on

line 1 are grouped after the L1 EQU in LOG1A. This construction is the same for all logical unit tables. Those devices which interrupt on line 2 are grouped after L2.

These logical unit tables are arranged to be parallel in structure and are indexed by logical unit number. The following apply to all logical unit tables.

Word 0 is always the maximum logical unit number or the table length-1

Word 1 is always the core allocator (the SPACE driver)

Other logical unit numbers are assigned according to the order in which the LOG1A is established.

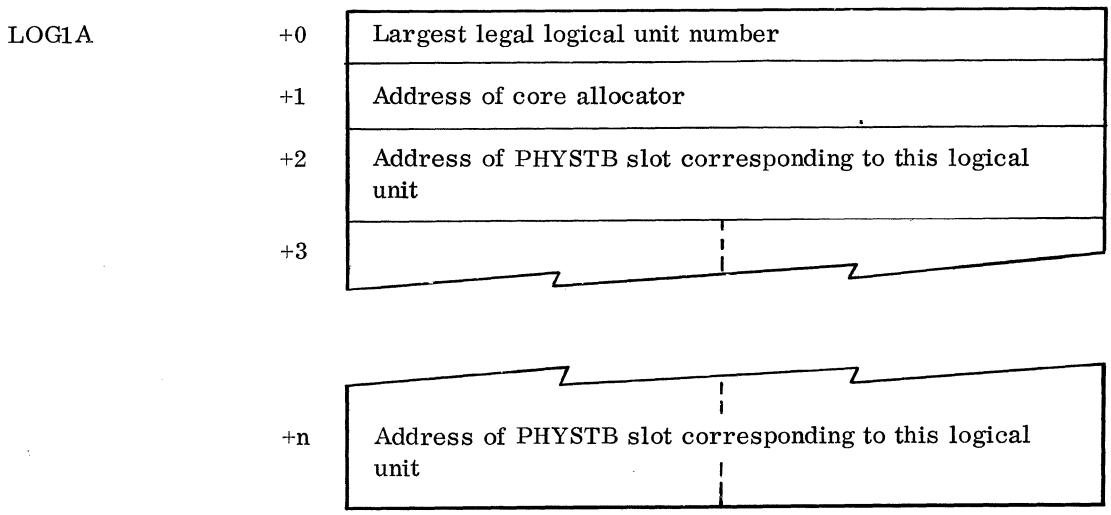
LOG1A

LOG1A contains the address constants of the PHSTB's. Each word in LOG1A contains the address of the first word of that logical unit's PHYSTB. Since there is a PHYSTB for each device, the next LOG1A word contains the address of the first word of the next PHYSTB.

When using EPROC, all physical devices are grouped according to the interrupt lines which they use. Therefore, all physical devices interrupting on line one are grouped after entry L1 in the LOG1A. But the logical unit numbers assigned to each of these units are determined by the order in which each of these are arranged within the entry.

When using a user-supplied interrupt response routine, instead of EPROC, the tags (L1, L2, etc.) are irrelevant; but the devices must still be in logical unit order.

LOG1A FORMAT



LOG1A Table and EPROC: If the LOG1A table is to be used with the external interrupt processor (EPROC), the following additional construction is necessary.

1. Group the devices by interrupt line number. This fixes the logical unit assignment.
2. Insert fifteen EQU statements of the form EQU Lx(*) (where x is a number from 1 to 15) in LOG1A. These EQU's are then used to identify the line number for the groups of devices. For example, EQU L1(*) precedes the device table addresses for the devices which interrupt on line 1. These are followed by EQU L2(*) and the device table addresses for the devices which interrupt on line 2, etc. To illustrate:

Label	Op	Address		
LOG1A	NUM	NUMLU		
	ADC	CORE	LU 1	
	EQU	L1(*)		
	ADC	PPTRDR	LU 2	interrupt line 1 devices
	ADC	PPTPCH	LU 3	
	ADC	TELPTR	LU 4	
	ADC	CARD29	LU 5	
	EQU	L2(*)		
	EQU	L3(*)		
	EQU	L4(*)		
	ADC	DISK0	LU 5	interrupt line 4 devices
	ADC	DISK1	LU 6	
	EQU	L5(*)		

3. Construct the SECPRO table (see SECPRO, Section III.1.1.3):

SECPRO	NUM	$7FFF, $7FFF, $7FFF, $7FFF, $7FFF, $7FFF
	NUM	$7FFF, $7FFF, $7FFF, $7FFF, $7FFF, $7FFF

Normally, all entries are left empty, i.e., $7FFF. The address of a special interrupt response routine may be included in the entry for its line, but it is more efficient to put this address in the fourth word of the interrupt trap location instead of using EPROC or SECPRO.

If EPROC is not used, the logical unit assignment numbers do not need to be equated to the interrupt lines.

To use a logical unit order which differs from the interrupt line order to which the peripheral devices are connected, use separate interrupt response routines.

LOG1

LOG1 is the alternate device table. Unless an alternate device or shared LUN is to be specified, entries in this table are initially set to 0. If an alternate device is to be assigned, set bits 9-0 to the alternate logical unit number.

If a device fails, the driver calls the alternate device handler with the logical unit of the failed device. The alternate device handler checks the LOG1 entry for this logical unit and if a nonzero alternate logical unit is found, the request is rethreaded on the alternate LUN and the driver for the alternate is scheduled to process the request. A message is also typed. If the alternate logical unit is out of service or has failed, the request is passed to the alternate of the alternate, etc. A message also appears. If no operational alternate exists, a request for operator intervention is made.

If two or more logical units share the same device table, set bit 14 of the corresponding LOG1 entry to 1.

The order of entries in the LOG1 is identical to that of the LOG1A.

LOG1 FORMAT

LOG1

Largest legal logical unit number						
15	14	13	12	11	10	9 Alternate logical unit number

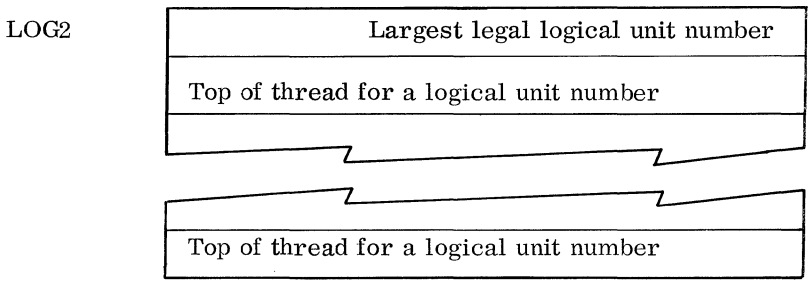
Bit	Significance	
15	0	Normal logical unit
	1	Buffer output logical unit
14	0	Logical unit does not share device with another logical unit
	1	Logical unit shares a device with another logical unit
13	0	Logical unit is operative
	1	Logical unit is out of service. Alternate, if any, is in use
12		Reserved
11	0	No operation
	1	If need to, restore logical unit on completion of buffer output request
10		Reserved
9-0		Alternate logical unit number should be set to the hexidecimal equivalent of the logical unit number

LOG2

LOG2 contains the top of thread for each logical unit. The order of entries in LOG2 is identical to the order of entries in LOG1.

Entries are initially set to $FFFF_{16}$.

LOG2 FORMAT



1.2.3 INTERRUPT MASK TABLE

MASKT is a table of M register interrupt line mask words which are arranged in the software priority level order. Only the monitor may change the M register. It uses the MASKT to set the M register according to the current priority level.

Standard MASKT

Most of the operating system programs have been assigned to the standard priority levels shown in the following table.

Level	System Program
-1	Idle loop
0	Job processor execution
1	Job processor I/O completion
2	Hang loop while a SWAP is in effect
3	Manual interrupt processor
4	Process programs
5	Process programs
6	Process programs
7	Core allocator
8	EOP for 1728-430 and 1729-2 card readers

Level	System Program
9	Disk, drum, and output message buffering package
10	Printer, paper tape punch, and paper tape reader
11	Magnetic tape drivers
12	Card reader, unbuffered magnetic tape
13	Timer interrupt and event counters; card reader
14	
15	Memory parity/protect fault routine

Construction and/or Modification of MASKT

The first step in constructing the MASK table is the assignment of software priorities. Follow these general concepts when developing the table.

1. Bits 0 through 15 of the M register correspond to interrupt lines 0 through 15. If, for example, bit 1 in the M register is set to zero, interrupts on interrupt line 1, the corresponding interrupt line, are locked out and are not processed until bit 1 in the M register is changed to a one.
2. Only the monitor can change the M register. It uses the MASKT to set the M register according to the current priority level.
3. Level -1 is used for the idle loop which must not include any monitor requests.
4. Each interrupt line normally has a 1 bit in the interrupt line position for all levels below the priority level associated with that line.
5. 0 bits must be placed in the interrupt lines position for all the priority levels equal to and above the priority level associated with the line.
6. Unused interrupt lines should be set to zero for each table entry.
7. More than one line can be associated with the same priority and can have the same mask.

Sample MASKT

PRIORITY LEVEL	INTERRUPT LINE 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	$MASK_{16}$
-1	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
1	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
2	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
3	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
4	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
5	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
6	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
7	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
8	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
9	0	0	0	0	0	1	0	0	0	0	1	0	1	1	1	1	042F
10	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	040D
11	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0405
12	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0005
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

Assembly language coding for this sample MASKT is:

Label	Op	Address	Comments
	NUM	$43F	PRIORITY LEVEL -1
MASKT	NUM	$43F	PRIORITY LEVEL 00
	NUM	$43F	PRIORITY LEVEL 01
	NUM	$43F	PRIORITY LEVEL 02
	NUM	$43F	PRIORITY LEVEL 03
	NUM	$43F	PRIORITY LEVEL 04
	NUM	$43F	PRIORITY LEVEL 05
	NUM	$43F	PRIORITY LEVEL 06
	NUM	$43F	PRIORITY LEVEL 07

Label	Op	Address	Comments
	NUM	$43F	PRIORITY LEVEL 08
	NUM	$42F	PRIORITY LEVEL 09
	NUM	$40D	PRIORITY LEVEL 10
	NUM	$405	PRIORITY LEVEL 11
	NUM	$05	PRIORITY LEVEL 12
	NUM	$01	PRIORITY LEVEL 13
	NUM	$01	PRIORITY LEVEL 14
	NUM	$00	PRIORITY LEVEL 15

1.2.4 VOLATILE STORAGE (VOLBLK)

Definition

VOLBLK is the volatile storage area which is primarily reserved for the allocation of small blocks of data storage for routines which are re-entrant (may operate at more than one level at the same time).

Allocation

Reserve enough volatile storage for each priority level to accommodate the maximum amount of volatile storage which could be requested at any one time because the system cannot recover from an overflow of volatile storage (i.e., requesting more storage than is available).

To compute allocation of volatile storage:

1. Allow 16 locations for each priority level making monitor requests. Eight of these locations are used for each request. The other eight locations may be used if the request processor itself makes a monitor request, such as the read/write request processor making a scheduler call for a driver.
2. Allow 88 locations (34 for locations $C5-$E5 and 54 for FLIST entry point address) for each priority level using the re-entrant FORTRAN library to allow the FORTRAN communications area and library subroutine entries to be saved.
3. Allow 57 locations for each priority level using the encode/decode package which is non-standard.

The following code defines volatile storage (see SYSBUF equivalences in section 1.2.1).

Label	Op	Address
VOLBLK	BSS	VOLBLK(16*NUMPRI+88*NFTNLV+57*NEDLVL+1)

1.2.5 INTERRUPT STACK AREA (INTSTK)

INTSTK is the block of storage which is set aside for saving the status of interrupted programs. The common interrupt handler stores the Q, A, I, and P registers and also the overflow indicator and the priority level of the interrupted program in this area. Five words are necessary for each entry. The stack is of the last-in, first-out type of stack on a priority basis.

The format of an entry is as follows.

	Word		
INTSTK	+0	Q register	Interrupted program running at priority level n
	1	A register	
	2	I register	
	3	Overflow (bit 15), P register	
	4	Priority level (=n)	
	5	Q register	Interrupted program running at priority level m
	6	A register	
	7	I register	
	8	Overflow (bit 15), P register	
	9	Priority level (=m)	

Level m< n

The following code defines the interrupt stack.

Label	Op	Address
INTSTK	BSS	INTSTK(5 * NUM PRI)

1.2.6 SCHEDULER STACK (SCHSTK)

A program requests the operation of another program by making a scheduler (SCHDLE) request. The timer routine can also make a SCHDLE request after a given interval of time has elapsed. These requests are threaded together on the scheduler thread.

The scheduler stack (SCHSTK) is a series of four-word entries.

Words one and two contain the scheduler call parameters (priority level and address of program scheduled).

Word three contains the thread to the next lower priority entry.

Word four contains the value of the Q register which is being passed to the requested program as a parameter.

The total number of entries required is equal to the sum of the number of scheduler requests and timer requests which can be in the stack at one time. The user may change the size of this stack. The scheduler stack normally has 25 entries.

Sample SCHSTK

Label	Op	Address	Comments
SCHSTK	ADC	0, 0, *+2, 0	LEVEL, COMPLETION ADDR., THREAD, Q REG.
	ADC	0, 0, *+2, 0	
		.	
		.	
		.	
		.	
	NUM	0, 0, -0, 0	LAST ENTRY
	EQU	SCHLNG (*-SCHSTK)	

1.2.7 ALLOCATABLE CORE (AVCORE)

EQU AVCOREnnnn is an entry in the SPACE program which defines the total size of the allocatable core area. CALTHD is the address of the location which contains the size of the first block which is initially all of allocatable core. Following this address is the address of the first piece (top of core allocator's thread) which is the beginning of the allocatable area.

No modification is necessary to the following code.

Label	Op	Address	Comments
CALTHD	ADC	AVCORE	NO. OF WORDS
	ADC	AREAC	ADDRESS

LVLSTR is the table of starting addresses for the allocatable core area available to each priority level. The upper bound for protected allocatable area is the same for all levels — the start of unprotected core. To prevent low priority programs from tying up all of the allocatable area, it is common to restrict the amount available to them while making the entire allocatable area available to the high-priority programs. Thus, a higher address usually appears for the low-priority programs.

Core swapping occurs at the following times.

A request for space is made at a request priority level greater than two

No unprotected I/O is in progress

A fixed interval of time has expired since the last swap

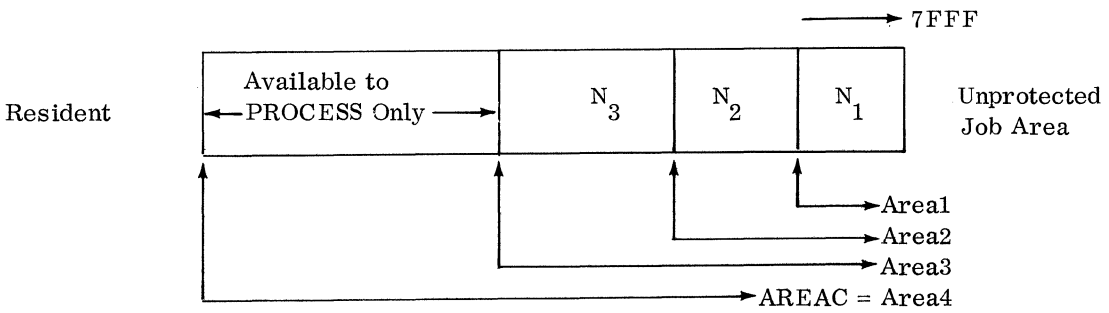
There is insufficient space available to that priority level in the allocatable area

Version 3.0 of the operating system automatically causes a core swap whenever job processing is terminated. This causes the job area (unprotected allocatable core) to be protected and made available to protected mass memory resident programs. The swapped condition continues until job processing is requested again by the operator.

Example of LVLSTR:

Label	Op	Address	Comments
	EXT	AREA1, AREA2, AREA3, AREA4	
LVLSTR	ADC	AREAC	0 REQUEST PRIORITY
	ADC	AREA1	1 LEVELS
	ADC	AREA2	2
	ADC	AREA3	3
	ADC	AREA4	4
	ADC	AREAC	5
	ADC	AREAC	6
		.	
		.	7-15
		.	
		.	
	ADC	LEND	

AREA1, AREA2, AREA3, and AREA4 are entry point names in the SPACE program used to divide the allocatable area, as shown in the diagram below (also refer to the SPACE program Section 1.3). Request priority levels 1, 2, and 3 include sufficient area for the job processor modules. The memory map for the LVLSTR table above is:



The entire allocatable core area (AREAC) must be available at request priority zero (RP equal to 0) so that the system may get started as initialized (job processor initiated with RP equal to 0 in system directory).

To make certain that the individual modules of the job processor can obtain sufficient allocatable core at all times, use the LIBEDT *S statements. Set the request priority for their system directory entries as follows. This operation is done after the operating system is built and is functional. The *S statements should be entered only once after the system is built, since the mass memory image of the system directory is actually updated by the *S.

*YM Entry Name	*YM Ordinal (Typical)	Request Priority
EFILE	1	0
LIBEDT	2	3
LOADSD	3	0
JOBENT	4	1
JOBPRO	5	2
PROTEC	6	3
JPLOAD	7	2
JPCHGE	8	2
JPT13	9	2
MIPRO	10	2
RESTOR	11	2
ODEBUG	12	3
RCOVER	13	3
BRKPDT	14	3
SELF	15	3
LOGGER	16	3

1.2.8 SPECIAL ROUTINES

IDLE

IDLE is the program which runs at level -1 when no other programs are running. This routine may be modified by the user. A counter may be included to compute the percentage of time spent at this level to provide a measure of the amount of idle time available in the main frame.

DUMMY

DUMMY is the dummy device driver. It is used with the dummy device table and is assigned a logical unit like a normal device. Read or write requests which address this logical unit cause the dummy driver to be initiated, and the completion address in the request is scheduled with error indication. This allows the dummy device to be set up as the alternate for devices where it would not be acceptable to hang up the request waiting for operator action in response to the alternate device handler request for input. This routine requires no modification.

FMASK, FLIST

FMASK and FLIST contain data for the re-entrant FORTRAN dispatcher and scheduler, RDISP. See section 3.4 (Re-entrant FORTRAN Library Package) and 3.5 (Non-re-entrant FORTRAN Library Package.) If the re-entrant FORTRAN library package and RDISP are used, FMASK and FLIST may require modification; if RDISP is not used, FMASK and FLIST may be removed. FMASK is a location which indicates the software priority levels which require the saving of the temporary area used by the FORTRAN routines. These levels must not also be assigned to interrupt lines since the interrupt handler does not save the FORTRAN data. A bit is set to 1 in FMASK in the bit position corresponding to each level using FORTRAN. If too many levels are allowed to run FORTRAN programs, the overhead for the low-priority programs may be unnecessarily high. For example, the following allows FORTRAN at levels 4, 5, and 6.

Label	Op	Address
FMASK	NUM	$0070

Levels 0 and 1 are reserved for unprotected programs and do not interrupt the priority levels using FORTRAN. Therefore, the mask is not set for levels 0 or 1.

FLIST Table

FLIST is the table of entry point locations in the FORTRAN library which must be saved in order to allow re-entrant use of the library. The symbolic names must also be declared as externals (EXT) and must appear as entry names (ENT) in the library subroutines.

Q8STP

Q8STP provides a branch to the dispatcher for FORTRAN object programs. It cannot be used by protected mass memory resident programs as a substitute for CALL RELESE main. The entry point name Q8STP is that generated by the compiler as an exit at the end of a compiled program.

CHRSFG

CHRSFG is a switch that indicates whether or not the on-line debug package (ODP) is running. When the debug package is running, CHRSFG is not zero.

NSCHED

NSCHED contains the maximum number of programs which may be scheduled per timer interrupt.

1.2.9 SPECIAL TABLES

Diagnostic Timer Table (DGNTAB)

DGNTAB is a table which consists of the PHYSTB addresses for all the devices to be supervised by the diagnostic timer program. Software buffer driver PHYSTB's may also be included in the table. The end of the table is indicated by a negative address, i.e., bit 15 = 1. Note that the first word in the table is not the table size.

To add a driver, place an entry in the diagnostic table. Each entry is a pointer to the physical device table for that device.

Example:

```
Label       Op      Address     Significance

            ENT     DGNTAB      DIAGNOSTIC TIMER TABLE
*
DGNTAB      ADC     CORE        1  CORE ALLOCATOR
            ADC     PPTRDR      2  1777 PAPER TAPE READER
            ADC     PPTPCH      3  1777 PAPER TAPE PUNCH
            ADC     TELPTR      4  1713 TELETYPE
            ADC     TPPDR1      6  601 MAG. TAPE UNIT 0
            ADC     TPPDR2      7  601 MAG. TAPE UNIT 1
            ADC     DISK0       8  853 DISK
            ADC     LP501       9  1740-501 LINE PRINTER
            ADC     CD1728     10 1728 CARD READER
            NUM     $FFFF         END OF TABLE
```

Alternate Device Handler (ALTERR)

ALTERR is the buffer table for the alternate device handler. It is used to save the error word (Q register) passed by a driver to the alternate device handler. Location ALTERR contains the table size, followed by a block of zeros of this size. The size should be set to the maximum number of simultaneous device failures possible. For most systems this equals the number of logical units.

1.2.10 MASS MEMORY DIAGNOSTIC ROUTINES (MMDIAG)

The routine MMDIAG is included in SYSBUF and is entered from either the drum or the disk driver in the event of a mass memory failure. The error code is passed in the Q register. The alternate device handler is not called from mass memory drivers since an alternate cannot be assigned and it may be desirable to attempt recovery after printing a diagnostic message.

MMDIAG is a routine which prints a message of the following form.

MASS MEM ERR code

The error code is from 0-11. For disk, see part II section 2.5; for drum, see part II, section 2.6

If the request which resulted in a failure was a system directory request, the routine releases the allocated core. Control then returns to the driver. Separate routines must be provided for systems with both drum and disk as MMDIAG is not re-entrant. The entry point names for these routines must be:

Label	Op	Address	Comments
	ENT	DMDIAG	DRUM
	ENT	DKDIAG	DISK

For disk or drum systems, remove the present EQU's which equate these entries to MMDIAG.

1.2.11 OVERLAY SUBROUTINE (OVRLAY)

The overlay subroutine, entry point OVRLAY, allows users to call for mass memory to be read over the actual call parameters. This is accomplished in the disk or drum drivers by moving the parameter list to the equipment table and using the OVRLAY subroutine to ensure that the return address from the call cannot be written over. Indirect overlay calls are not permitted. The overlay subroutine may be removed if no overlay calls are included in the system. The basic operating system, the Macro Assembler, and the FORTRAN compiler do not use the overlay subroutine.

1.2.12 PHYSICAL DEVICE TABLE (PHYSTB)

Each physical device has a PHYSTB (physical device table) which contains all device data necessary for a device to be operated by its driver. Generally this data includes:

Entry addresses to the driver responsible for operating the device

Equipment word telling the driver which device to use

Information which allows the driver to fulfill the current request

The table contains at least 15 words for each device. Words 0 through 15 have a standard function for all devices. Words 15 on are used for special purposes for each driver. The system programmer should remove the device tables which are not needed for a particular system. If additional device tables are needed, use the existing device tables as a guide. However, normally make only the following changes:

The label on word 0 (ℓ)

The equipment address in word 7

Occasionally, when a driver must drive several devices, a word in the PHYSTB is used to thread one PHYSTB to another

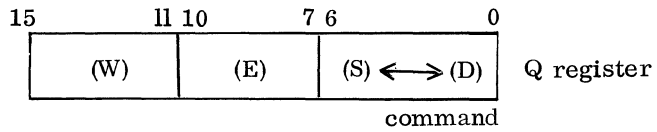
The hardware type in bits 10 through 4 in word 8.

PHYSICAL DEVICE TABLE FORMAT (PHYSTB)

WORD	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	SYMBOLIC NAME	USE
ℓ 0	0	0	0	1	0	0	1	0	0	0	0	0					ELVL	STANDARD FOR ALL DRIVERS
1	DRIVER INITIATOR ADDRESS																EDIN	
2	DRIVER CONTINUATOR ADDRESS																EDCN	
3	DRIVER I/O HANGUP DIAGNOSTIC ADDRESS																EDPGM	
4	DIAGNOSTIC CLOCK ADDRESS																EDCLK	
5	DEVICE LOGICAL UNIT																ELU	
6	CURRENT REQUEST PARAMETER LIST ADDRESS																EPTR	
7	CONVERTER CODE						EQUIP-MENT CODE				STATION CODE						EWES	
8																	EREQST	
9																	ESTAT1	
10	CURRENT BUFFER ADDRESS																ECCOR	
11	LAST WORD ADDRESS + 1 OF BUFFER																ELSTWD	
12	LAST EQUIPMENT STATUS READ																ESTAT2	
13	DRIVER LENGTH IF MASS MEMORY																	USE WHEN REQUIRED BY DRIVERS OR FOR THE OUTPUT MESSAGE BUFFERING PACKAGE
14	NAME ASSOC. WITH SECTOR NUMBER																	
15																		

Word	Bit	Significance
0		ELVL $120x A SCHDLE request to operate the driver initiator address at level x. x is the initiator priority level which should equal the priority level of the interrupt in the LOCORE program.
	14-9	Request code for SCHDLE request.
	8-4	Unused, unless specified by a particular driver.
	3-0	Priority level at which driver operates.
1		EDIN Driver initiator address (which is the second word of the SCHDLE request).
2		EDCN Driver continuator address. Control is transferred to this address (when interrupt occurred) at the priority level assigned to the interrupt in the interrupt trap region. This priority level must be the same as the priority level specified by word 0.
3		EDPGM Driver error routine address. Control is transferred to this address at the driver priority level when the diagnostic clock is counted down to negative by the diagnostic timer.
4		EDCLK Diagnostic clock. This diagnostic clock location is set by the driver and decremented by the diagnostic timer for a hardware completion interrupt. Set idle (-1) by Complete Request Routine.
5		ELU Logical unit currently assigned to device. 0 if device not in use. Set by request processor; may be reassigned by FNR routine; cleared by the next FNR routine or complete request.
6		EPTR Address of caller's parameter list. Set by FNR routine.

Word	Bit	Significance
7		EWES Hardware address. To obtain equipment status: load this word into into the Q register and perform INPUT instructions. Status is saved in ESTAT2, word 12. See Control Data 1700 Computer System Code 60163500.



15-11	(W) Coverter code Code 1706 2 #1 7 #2 C #3 0 When coupled directly to AQ channel
10-7	(E) Equipment code. Equipment numbers for the released operating system drivers are listed in part III, section 2. Suggested equipment codes for additional drivers are in part II, section 2 along with the information for each driver.
6-0	Command code. The command code is divided into two sections: S contains the station code and D contains the director function. The station code is located in bit 6 and adjacent lower order bits as required. The director function is located in bit 0 and adjacent higher order bits as required. They cannot overlap and all bits in the command code are not necessarily used. If the controller does not contain any stations, the station code is zero.

Word	Bit	Significance
8		EREQST Request status.
	15	Busy bit. 0 Operation complete 1 Operation is in progress
	14	0 If no device error 1 If driver detects device failure
	13-11	Equipment class code 0 Class not defined 1 Magnetic tape device 2 Mass storage device 3 Card device 4 Paper tape device 5 Printer device 6 Teletype device 7 Reserved for future use
	10-4	Numbers in the following list are in decimal and must be converted to hexidecimal before inserting in bits 10 through 4. Equipment type code (T). 0 1711/1712 teletypewriter 1 1777 paper tape reader 2 1777 paper tape punch 3 Unassigned 4 Unassigned 5 1738-853 disk unit 6 1751 drum unit 7 Unassigned 8 1738-854 disk unit 9 601 magnetic tape unit 10 Software buffering device 11 Unassigned 12 1728-430 card reader/punch 13 Software core allocator 14 210 CRT display station 15 1558 latching relay output 16 1553 external register output 17 311B/312B data set terminal 18 322/323 teletype terminal 19 501 line printer 20 166 line printer 21 1612 line printer 22 415 card punch 23 405 card reader 24 608 magnetic tape unit 25 609 magnetic tape unit

Word	Bit	Significance
		26 1713 teletype keyboard
		27 1713 TTY paper tape punch
		28 1713 TTY paper tape reader
		29 1729-2 card reader
		30 1797 buffered I/O interface
		31 Software dummy alternate
		32 1584 selectric I/O typewriter
		33 1582 flexowriter I/O typewriter
		34 1716 compiling data channel
		35 1718 satellite coupler
		36 Unassigned
		37 8000 series magnetic tape unit
		38 1732-608 driver
		39 1732-609 driver
		40 1530 A/D converter 30/40 PPS
		41 1534 A/D converter 200 PPS
		42 1538 A/D converter high speed
		43 Unassigned
		44 Unassigned
		45-99 Reserved for future standard equipment
		100-127 Open for user assignment
	3	0 PHYSTB does not contain message buffering in words 18-33. 1 PHYSTB includes words 18-33 for message buffering.
	2	0 Device may not be written by unprotected programs 1 Device may be written by unprotected programs
	1	0 Device may not be read from unprotected programs 1 Device may be read from unprotected programs
	0	0 Device available to unprotected programs 1 Device not available to unprotected programs
9		ESTAT1 Status word 1.
	15	0 No error occurred 1 If error condition and/or end-of-file detected by driver
	14	0 If the number of words which were requested were transferred on a read request 1 Set by driver if fewer words were read than requested
	13	0 No end-of-file is sensed 1 Set by driver if device remains ready after detecting an error or end-of-file or both

Word	Bit	Significance
	12	Reserved for message interpreter request
	11	0 No error 1 Set by output message buffering package if message buffer output is incomplete
	10	0 No parity error occurred 1 Set by driver if parity error occurred
	9	Reserved
	8	Reserved for individual drivers' special use
	7	Reserved for individual drivers' special use
	6	Reserved for individual drivers' special use
	5	Data control word indicator: 0 This is not a control character 1 Set by driver if this is a control character
	4	First character of FORMAT record set by driver 0 This is not first character 1 Set by driver if this is first character
	3	Mode set 0 Set by driver when binary mode is used 1 Set by driver when ASCII mode is used
	2	Case indicator 0 Set by driver if this is lower character 1 Set by driver if this is upper character
	1	Format read/write indicator set by FNR routine 0 Unformatted record read/write 1 Formatted read or write request
	0	Read/write indicator set by FNR routine 0 Read request 1 Write request
10		ECCOR The driver will store or obtain next data from this location which was initially set by FNR routine but is updated by the driver.
11		ELSTWD The driver will satisfy the request by either storing or obtaining from this location which is the last data location +1.
12		ESTAT2 Status word 2. The last value of equipment status mentioned in word 7.

Word	Bit	Significance
13		Set to 0 if the driver is core resident. Set to the length of the driver if the driver is mass memory
14		Name associated with the sector number
15		Temporary storage and return from subroutines (FNR)
16 and beyond		Used when required by drivers or for the output message buffering package

1.2.13 INTERRUPT RESPONSE ROUTINE

Single Device Interrupt Lines

The following example is typical of an interrupt line which serves only one device.

Word	Label	Op	Address	Comments
		END	I1728	
	I1728	LDQ	=XCD1728	PHYSTB ADDRESS
		JMP*	(CD1728+2)	GO TO CONFIGURATOR

The addresses of the interrupt response routines must be declared as entry names, since they are externals in LOCORE.

CAUTIONS

For some devices the status check may need to be coded differently.

Some drivers may not need a multiple device interrupt response routine. If the driver can address only one device at a time, it saves the address of the PHYSTB for the last device addressed.

Some interrupts are not associated with drivers (e.g., 1573 timer), and the interrupt response is an integral part of the program that handles the device.

1.3 SPACE

The SPACE program includes the SPACE request processor, the allocatable core area, and the restart program. No modification is needed to the space request processor. The allocatable core area should be customized for each system.

1.3.1 ALLOCATABLE CORE

AREAC is the start of the block of allocatable core within which the mass memory resident programs are executed. The total area available is specified by the following:

```
Word    Label   Op      Address                 Comments
        AREAC   ADC     AVCORE
                ADC     ($7FFF)
                EQU     N1($19F)                (Area 1)
                EQU     N2(1B0)                 (Area 2)
                EQU     N3($300)                (Area 3)
                EQU     N4($10A)                (Area 4)
                BSS     (N4-INPUT+RESTRT-1)
                BSS     AREA3(N3+2)             } Reserves the desired core
                BSS     AREA2(N2+2)             } areas and defines the
                BSS     AREA1(N1+2)             } starting address of
                BSS     (2)                     } each area
                EQU     AVCORE(*-AREAC)
                EQU     AREA4(AREAC)
                ENT     AREA1, AREA2, AREA3, AREA4
```

1.3.2 RESTART PROGRAM (RESTRT)

RESTRT is the starting address of the restart program which is entered from the AUTOLOAD program. The system initializer builds the AUTOLOAD program during initialization and places it on the first sector (96 words) of mass memory. After transferring the image of the protected programs into core, control passes to RESTRT via location 1 in the communications region.

The MSOS 3.0 restart program also includes provision to start the 1572/1573 timer and to schedule the diagnostic timer program. If the timer is not present or if it is switched off, a reject occurs and the message TIMER RJ is written on the comment device. The program requests the monitor to type PP and waits for the operator to acknowledge the setting of the PROGRAM PROTECT switch by typing an asterisk followed by pressing RETURN. Note that a monitor request is used to type PP. If autoload does not result in PP being typed, the monitor probably was not set up properly.

Since the restart program is only used immediately after an autoload, it executes in the allocatable area, but it is set up as though it were part of the core-resident programs. In this way, the program does not require any permanent core storage, and it is destroyed as soon as a mass memory resident program is scheduled.

Modification of the restart program may be desired to allow initialization of data to occur after autoload without providing permanent core for such an initialization program. For example, code to start a process may be inserted here. Such additions may only be added just prior to the request to type PP.

1.4 ENGINEERING FILE

The engineering file is a permanent log of hardware errors occurring on a device.

When a user's configuration deviates from the released version, the engineering file must be customized. The program EF on mass storage establishes the engineering file. The order and the size of the file depends upon the order and number of logical units.

All devices except mass storage devices have an engineering file on mass storage. Even though the engineering file for mass memory devices is not kept on mass storage, space for each logical unit must be reserved in the correct logical unit position.

To customize the engineering file, assemble the program EFILE on the COSY (MSOS 3.0) tape. Each entry in the file requies 21 words:

Label	Op	Address	Significance
	ALF	4, LU NAME	
	BZS	(17)	

The engineering file allows eight characters for a symbolic name for each device and 17 words for the file itself. For a correct file printout, the symbolic name must coincide with the user's configuration.

Since the engineering file for mass storage devices is kept in core in the alternate device handler, customize the alternate device handler (ADEV) also. The equate in ADEV depends upon the number of mass memory devices. The necessary code is:

```
ADEV    DCK/    I=lu, H=lu
        DEL/    440
        EQU/    MASNUM(x)       x=number of mass memory devices
        END/
```

DRIVER ADDITION 2

To insert a driver, the following tables described in the sections of part II specified must be modified or added:

	Section
Logical unit table LOG1A, LOG1, LOG2	1.2.2
Interrupt mask table (MASKT)	1.2.3
Diagnostic timer table (DGNTAB)	1.2.9
Physical device table (PHYSTB)	1.2.12
Interrupt response routines	1.2.13
Interrupt trap area	1.1.3
Engineering file	1.4

Changes unique to each driver are outlined in this section. For additional information, such as table structures, see the sections in part II referenced above.

Driver deletion is described in step 5 of part I, section 3. Special information on the 1706 buffered data channel (section 2.1) and special instructions for any driver to be installed on mass memory (section 2.15) are also in this section.

Standard Installation I/O Capabilities

The drivers incorporated on the standard install tape allow formatted read and writes and/or unformatted read and writes on all devices except on magnetic tape. The magnetic tape drivers only allow formatted I/O. To have unformatted I/O for magnetic tape, add RWBA and delete *S, RWBA, 7FFF from the install tape.

Outline of Section 2

Those drivers preceded by an asterisk in the list below are already on the released system tape.

2.1 1706 BUFFERED DATA CHANNEL

The 1706 buffered data channel may be used to buffer any of the following devices:

1726-405 card reader

1732-608/609 magnetic tape unit

1731-601 magnetic tape unit

Installation procedures are:

1. Assemble the driver and obtain a relocatable binary

   a. If buffering the 1726-405, assemble CR405, setting the equate to 1:

Label	Op	Address	Comments
CR405	DCK/	I=lu, H=lu	
	DEL/	138	
	EQU	BUFER (1)	
	END/		

   b. If buffering the 1731-601, assemble the following modules:

      TAPDRB

      RWBAB

      FRWAB

      FRWBB

      RECVTB

      TAPE

   c. If buffering the 1732-608/609, assemble DR1732, setting the P2 parameter to BUF

2. Assemble the BUFALC module, setting the equate to the number of devices on each converter.

BUFALC	DCK/	I=lu, H=lu	
	DEL/	38,40	
	EQU	NDEV1(x)	Number of devices on converter 1
	EQU	NDEV2(x)	Number of devices on converter 2
	EQU	NDEV3(x)	Number of devices on converter 3
	END/		

3. SYSBUF: Insert a PHYSTB using the information in part II, section 1.2.12. Set bit 12 of word 7 to 1.

4. SYSBUF: Insert an interrupt response routine similar to the following:

   a. 1726-405

```
Label       Op      Address
            ENT     INT405
INT405      LDQ     =XCR405
            JMP     (CR405+2)
```

   b. 1731-601

```
            ENT     IN601
INT601      LDQ     =XTPPDR1
            JMP     (TPPDR1+2)
```

   c. 1732-608/609

```
            ENT     TPBUSY
TPBUSY      ADC     0
            ENT     TPINT
TPINT       LDQ*    TPBUSY
            LDA-    2,Q
            STA-    I
            JMP-    (ZERO),I
```

5. LOCORE: Assemble LOCORE setting up the interrupt trap area with the appropriate level and interrupt response routine address. In the example below, x is the priority level; y is the interrupt response routine name. All buffered drivers must run at the same priority level.

```
LINEx       NUM     0
            RTJ-    ($FE)
            NUM     x
            ADC     y
            EXT     y
```

2.2 1726-405 CARD READER DRIVER

2.2.1 DESCRIPTION

The 1726-405 card reader driver allows data input from the 405 card reader to core. This driver may be installed mass memory or core resident.

2.2.2 INSTALLATION REQUIREMENTS

SYSBUF

System tables and parameters	3
Physical device table	107
Diagnostic timer table	1
	111

Driver Requirements

Hardware conversion non-buffered	351
Hardware conversion buffered	384
Software conversion non-buffered	451
Software conversion buffered	464

2.2.3 INSTALLATION PROCEDURES

1. Insert an interrupt entry similar to the following into the appropriate interrupt trap area of LOCORE. Priority 8 is used for this example.

Label	Op	Address
LINEx	NUM	0
	RTJ-	($FE)
	NUM	8
	ADC	INT405

2. Insert in LOG1A after EQU Lx(*)

Label	Op	Address
	ADC	CR405

3. Insert in LOG1:

	ADC	0

4. Insert in LOG2:

	NUM	$FFFF

5. If the system has a timer package, insert in the diagnostic timer table:

	ADC	CR405

6. Modify the MASKT according to part II, section 1.2.3.

7. Installation tape:

   a. Replace LOCORE and SYSBUF with the newly updated versions

   b. Insert BUFALC as a core resident module after an *L statement

   c. Insert the driver as core resident or as mass memory

8. Construct a PHYSTB for the 1726-405 using the following instructions and sample PHYSTB as a guideline:

   a. Declare the driver entry point names as external

   b. Word 0: select the priority level of the scheduler request so that it corresponds to the priority level selected in the appropriate interrupt trap area of LOCRE (step 1). The sample PHYSTB below uses priority level 8

   c. Words 1, 2, 3: determine the addresses of the initiator, continuator, and the error routine

   d. Word 7: select the hardware connect address. The sample PHYSTB which follows uses 0201 which is derived from using equipment number 4 and directory function 1

Word	Label	Op	Address	Significance
		EXT	IN1726, CN1726,EX1726, MAS405	
0	CR405	NUM	$1208	
1		ADC	IN1726	
2		ADC	CN1726	
3		ADC	EX1726	
4-6		NUM	-1,0,0	
7		NUM	$0201	Equipment word
8		NUM	$1972	Request status
9-12		NUM	0,0,0,0	
13		NUM	0	Driver length if mass memory
14		NUM	MAS405	Name associated with sector number
15		NUM	0	
16		NUM	$6	EOF card pattern
17		ADC	BF1726	Starting address of 80 word buffer
18-26		BZS	(9)	
27		BZS	BF1726(80)	80 word buffer

9. Several assembly options are available. The released version of the 1726-405 driver specifies an unbuffered system using ASCII 1963. To specify different options, change the EQU BUFER and the EQU ASCI68 COSY cards. Change these cards before assembly and then make a new binary tape. Following are the possible options:

EQU	BUFER(0)	Unbuffered driver
EQU	BUFER(1)	Buffered driver
EQU	ASCI68(0)	Driver which converts ASCII 1963
EQU	ASCI68(1)	Driver which converts ASCII 1968
EQU	ASCI68(2)	Driver which converts ASCII 1968 with CDC subset
EQU	MM(0)	Driver is core resident
EQU	MM(1)	Driver is mass memory

10. Interrupt response routine for both the buffered and unbuffered driver.

Label	Op	Address
	ENT	INT405
INT405	LDQ	=XCR405
	JMP	(CR405+2)

11. Install the driver under a *L statement.

12. If installing on mass memory, see part II, section 2.15.

2.3 1728-430 READER-PUNCH DRIVER

2.3.1 DESCRIPTION

The 1728-430 reader-punch driver executes at high priority while data is read in and at low priority while data is interpreted, converted and packed. This minimizes possible destructive interactions with other concurrently executing drivers.

2.3.2 INSTALLATION REQUIREMENTS

Core Memory

Driver	866
PHYSTB and buffer	107
System tables	4
	977

2.3.3 INSTALLATION PROCEDURES

1. Use the Macro Assembler to assemble the 1728-430 driver routine and to produce a relocatable binary tape.

2. Following is an example of an interrupt trap which must be inserted with x as the interrupt line on which the 1728-430 is to be connected:

```
Label     Op      Address
LINEx     NUM     0
          RTJ-    ($FE)
          NUM     13
          ADC     I1728
          EXT     I1728
```

3. Declare the following names as external and entry symbols in SYSBUF anywhere before END and after NAM:

```
          EXT     IN1728
          EXT     CN1728
          EXT     EX1728
          EXT     MAS28

          ENT     I1728
```

4. Insert into LOG1A the label associated with word 0 of the PHYSTB, x is the interrupt line to be connected to the 1728-430.

```
          EQU     Lx(*)
          ADC     label
```

5. An interrupt response routine is advisable for the card reader to save time. The following will suffice:

```
I1728     LDQ     =XCD1728
          JMP*    (CD1728+2)
```

6. Add a zero cell to LOG1 at the logical unit position corresponding to the 1728-430 entry made in LOG1A using the following form:

```
          ADC     0
```

7. Add to LOG2 the following code at the logical unit position which corresponds to the 1728-430 entry made in LOG1A:

```
NUM     $FFFF
```

8. Modify MASKT according to instructions in part II, section 1.2.3.

9. Insert the following PHYSTB consisting of 27 words after the last PHYSTB inserted in the system. After word 26, insert the 80 word buffer. Place a label on word 0 to match the LOG1A entry.

Word	Label	Op	Address	Significance
0	CD1728	NUM	$120D	
1		ADC	IN1728	Initiator entry
2		ADC	CN1728	Continuator entry
3		ADC	EX1728	Error entry
4-6		NUM	-1,0,0	
7		NUM	$0421	
8		NUM	$08C6	Magnetic tape equipment type to allow motion control requests
9-12		NUM	0,0,0,0	
13		NUM	0	Driver length if mass memory
14		ADC	MAS28	Name associated with sector number
15		NUM	0	
16		ADC	BF1728	
17-26		BZS	(9)	
		BZS	BF1728(80)	

10. If installing on mass memory, see part II, section 2.15.

2.4 1729-2 CARD READER DRIVER

2.4.1 DESCRIPTION

The 1729-2 card reader executes at high priority while data is read in and executes at low priority while data is interpreted, converted, and packed. This minimizes possible destructive interaction with other concurrently operating drivers.

2.4.2 INSTALLATION REQUIREMENTS

Core Memory

Driver	440
PHYSTB and buffer	107
System tables	4
	551 words of core memory

2.4.3 INSTALLATION PROCEDURES

1. With x as the interrupt line on which the 1729-2 is to be connected, insert the following in the interrupt trap area of LOCORE:

```
Label     Op      Address
          EXT     I1729
LINEx     NUM     0
          RTJ-    ($FE)
          NUM     13
          ADC     I1729
```

2. Insert the following names as external symbols in SYSBUF anywhere between NAM and END:

```
          EXT     EX1729
          EXT     IN1729
          EXT     CN1729
          EXT     MAS292
```

3. Enter into LOG1A the label associated with word 0 of the PHYSTB:

Label	Op	Address
	ADC	label

4. Insert in LOG1 a zero cell at the index position corresponding to the 1729-2 entry made in LOG1A:

	ADC	0

5. Add to the LOG2 at the logical unit position corresponding to the 1729-2 entry made in LOG1A:

	NUM	$FFFF

6. Modify the MASKT according to the instructions in part II, section 1.2.3.

7. Insert this PHYSTB with a label on word 0:

Word	Label	Op	Address	Significance
0	CD1729	NUM	$120x	x is the initiator priority level which should equal the priority of the interrupt line in program LOCORE.
1		ADC	IN1729	
2		ADC	CN1729	
3		ADC	EX1729	
4-6		NUM	-1,0,0	
7		NUM	$621	Equipment word (equipment C)
8		NUM	$1C72	Equipment type
9-12		NUM	0,0,0,0	
13		NUM	0	Driver length is mass memory
14		ADC	MAS292	Name associated with sector number
15		NUM	0	
16		NUM	$xxxx	xxxx represents the 12 bits of column one which are to be interpreted as an end file card. For example: $0006 would mean that 8 punch is an end of file.

Word	Label	Op	Address	Significance
17		ADC	BF1729	BF1729 is the address of an 80 word BZS in the SYSBUF program.
18-26		BZS	(9)	
27		BZS	BF1729(80)	

If the driver is to be mass memory, word 13 of the PHYSTB is as follows:

	NUM	$1CC	Driver length

8. Enter the following interrupt response routine into SYSBUF:

```
        ENT     I1729
I1729   LDQ     =XCD1729
        JMP*    (CD1729+2)      Exit to the continuator
```

2.5 1738-853/854 DISK DRIVER (DISKWD)

2.5.1 DESCRIPTION

The 1738-853/854 disk driver provides the capability for data transfer to and from the disk as a mass memory device. Additionally, the disk driver handles the transfer of mass memory resident programs into core as a result of SCHDLE requests. This driver permits word addressability simulation.

The 1738 disk interface uses the direct storage access bus for its input/output to provide completely buffered operation. The disk driver complements this capability by requiring control upon end-of-operation of error condition only as indicated by an interrupt.

2.5.2 INSTALLATION REQUIREMENTS

Mass Memory

None

Core Memory

Core requirements for one disk driver are:

Driver	425 words
System tables and parameters	
Logical unit tables	3 words
Physical equipment table	22 words
Interrupt response routine	3 words
OVRLAY subroutine	8 words
Interrupt trap region	4 words
	465 words

Core requirements for a two disk driver are the same as for one disk driver except for the following deviations. Logical unit tables require a total of 6 words and the two PHYSTBs require a total of 44 words.

2.5.3 INSTALLATION PROCEDURES

1. Insert the following four-word interrupt entry using the desired interrupt line in place of x.

```
Label     Op      Address
LINEx     NUM     0
          RTJ-    ($FE)
          NUM     9
          ADC     EPROC
```

2. Insert in LOG1A an entry for each disk.

   Using one disk

```
          ADC     DISK0
```

   Using two disks

```
          ADC     DISK0
          ADC     DISK1
```

3. Insert in LOG1 an entry for each disk.

   Using one disk

Label	Op	Address
	ADC	0

   Using two disks

	ADC	0
	ADC	0

4. Insert in LOG2 an entry for each disk.

   Using one disk

	NUM	$FFFF

   Using two disks

	NUM	$FFFF
	NUM	$FFFF

5. Insert the following coding in SYSBUF:

	ENT	DISK0
	EXT	DKINTR, DKCONT, DKDIAR

6. Insert the following PHYSTB when installing one disk driver:

Word	Label	Op	Address
0	DISK0	NUM	$1209
1		ADC	DKINTR
2		ADC	DKCONT
3		ADC	DKDIAR
4-6		NUM	-1, 0, 0
7		ADC	$181

Word	Label	Op	Address	Comments
8		ADC	$xxxx	$1086 if using the 853 $1096 if using the 854
9		ADC	$200	
10-15		BZS	(6)	
16		ADC	$11A	
17		ADC	0	
18-21		BZS	(4)	18 to 21 overlay calls moved to here

Insert the following PHYSTB for two disks per controller:

0	DISK0	NUM	$1209	Library and scratch disk
1		ADC	DKINTR	
2		ADC	DKCONT	
3		ADC	DKDIAR	
4-16		NUM	-1, 0, 0, $181, $1056, $200, 0, 0, 0, 0, 0, 0, $11A	
17		ADC	DISK1	
18-21		BZS	(4)	Words 18 to 21 are for overlay calls moved here
	*			
	*			
0	DISK1	NUM	$1209	
1		ADC	DKINTR	
2		ADC	DKCONT	
3		ADC	DKDIAR	
4-16		NUM	-1, 0, 0, $181, $1056, $200, 0, 0, 0, $1000, 0, 0, $31A	
17		ADC	DISK0	
18-21		BZS	(4)	Words 18-21 overlay calls moved here

7. If the time-out surveillance is desired, insert the following entries into the diagnostic timer table.

   For one disk

Label	Op	Address
	ADC	DISK0

   For two disks

	ADC	DISK0
	ADC	DISK1

8. DKDIAG, the disk diagnostic subroutine, resides in SYSBUF and handles all error recovery for the driver. If the user wishes to supply his own routine, that routine must comply with the following requirements which apply to the present standard version.

   If a read of a system directory program into allocated core produces an error, this allocated core must be released.

   It can take diagnostic action as desired.

   It is a closed subroutine which is entered by a RTJ instruction.

   It must be entered with the disk PHYSTB address in the I register to allow access to the request parameters for diagnostic action.

   The standard version types the message:

   MASS MEM ERR code

   0 Time-out error; 1738 malfunction; no completion interrupt occurred as a result of disk operation initiation.

   1 Internal or external reject occurred on an INP or OUT instruction. Possible causes are: equipment turned off, erroneous equipment code, or 1738 malfunction

   2 Alarm

   3 Parity error

   4 Checkword error

   The nature of the error is also indicated in the Q register upon entry to DKDIAG with one of the codes listed above. If an error occurs on a SCHDLE request, the assigned core area is released; no completion address is scheduled. When DKDIAG is finished, it returns control to the driver with the I register intact.

2.6 1751 DRUM DRIVER

2.6.1 DESCRIPTION

The 1751 drum driver (DRMDRZ) provides a capability for data transfer to and from the drum as a mass memory device. Additionally, the drum driver handles the transfer of mass memory resident programs into core as the result of SCHDLE requests.

The 1751 drum interface uses the direct storage access bus for its input/output to provide completely buffered operation. The drum driver complements this capability by requiring control only upon end-of-operation of error condition as indicated by an interrupt.

2.6.2 INSTALLATION REQUIREMENTS

Mass Memory

None

Core Memory

Driver	272 words
System tables and parameters	
Logical unit tables	3 words
Diagnostic time-out (DGNTAB)	1 word
Physical equipment table	38 words
Interrupt response routine	3 words
OVRLAY subroutine	8 words
Interrupt trap region	4 words
Diagnostic subroutine (DMDIAG)	33 words
	362 words

2.6.3 INSTALLATION PROCEDURES

1. Insert the four-word interrupt entry which is associated with the drum interrupt line. It must contain the following, with x as the particular interrupt line to be used.

```
Label       Op      Address
LINEx       NUM     0
            RTJ-    ($FE)
            NUM     10
            ADC     EPROC
```

2. Into LOG1A, enter:

```
            ADC     DRUM
```

3. Enter into LOG1:

```
            ADC     0
```

4. Into LOG2, enter:

```
            NUM     $FFFF
```

5. Insert in SYSBUF:

```
            ENT     INTDRUM
            EXT     DRMINT, DRMCON, DRMERR
```

6. Insert the interrupt response routine in SYSBUF:

```
INTDRM      LDQ     =XDRUM
            JMP*    (DRUM+2)
```

7. Add the following to SYSBUF:

```
            EQU     E(2)
```

8. When adding the drum PHYSTB, the driver priority level is 10, the equipment code is 2, the coding is as follows:

Word	Label	Op	Address
0	DRUM	NUM	$12AA
1		ADC	DRMINT
2		ADC	DRMCON
3		ADC	DRMERR
4-6		NUM	-1,0,0
7-9		ADC	E*$80+1,$1066,$200
10-12		ADC	0,0,0
13-18		BSS	(6)
19-20		ADC	0,E*$80+$8
21-22		ADC	0,E*$80+$A
23-24		ADC	0,E*$80+$C
25-26		ADC	0,E*$80*$E
27-28		ADC	8,E*$80+$1
29-30		ADC	0,0
31-32		ADC	0,-1
33-35		NUM	4,0,0
36-37		ADC	E*$80,E*$80+$4

9. Add the drum overlay subroutine in SYSBUF:

```
OVRLAY  0       0
        IIN     0
        LDA*    OVRLAY
        ADD-    $32
        STA*    ORVL1
        RTJ-    ($F4)
OVRL1   0       0
        JMP-    ($EA)
```

10. If time-out surveillance is desired, add the following entry to the DGNTAB:

Word	Label	Op	Address
		ADC	DRUM

11. DMDIAG, the drum diagnostic subroutine, resides in SYSBUF and handles all error recovery for the driver. If the user wishes to supply his own routine, that routine must comply with the following which apply to the present standard version:

    If a read of a system directory program into allocated core produces an error, this allocated core must be released.

    It can take diagnostic action as desired.

    It is a closed subroutine which is entered by a RTJ instruction.

    It must be entered with the drum PHYSTB address in the I register to allow access to the request parameters for diagnostic action.

    The standard version types the message: MASS MEM ERR code.

Code	Significance
0	Time-out error; no completion interrupt occurred as a result of initiation of a drum operation; 1751 malfunction
2	Request not successfully completed because of occurrence of an irrecoverable error condition, defined previously
3	Timing synchronization error occurred while the drum was busy
4	Internal reject
5	External reject
6	Timing synchronization error occurred while the drum was busy
7	Not used
8	The request completion address parameter C lies within the range of a transfer not completed because of an irrecoverable error. This condition normally occurs as the result of a SCHDLE OVRLAY request, and DMDIAG is responsible for releasing core assigned by the SPACE request processor, if desired
9	Guarded address error on a WRITE or FWRITE request

    The nature of the error is also indicated in the Q register upon entry to DMDIAG with one of the codes above. When DMDIAG is finished, it returns control to the driver with the I register intact.

2.7 1740-501 LINE PRINTER DRIVER

2.7.1 DESCRIPTION

The 1740-501 line printer driver allows data output from core memory to the 501 line printer.

2.7.2 INSTALLATION REQUIREMENTS

Driver	511 words
System tables and parameters	6 words
PHYSTB	20 words
Diagnostic timer table	2 words
	539 words of core memory

2.7.3 INSTALLATION PROCEDURES

1. Insert an entry similar to the following into the appropriate interrupt trap area of LOCORE:

Label	Op	Address	Significance
LINEx	NUM	0	Entry
	RTJ-	($FE)	Interrupt handler
	NUM	10	Priority level
	ADC	EPROC	Interrupt response routine

2. Insert in LOG1A after EQU Lx(*):

SLO	ADC	LP501	Non-FORTRAN
FLP	ADC	LP501	FORTRAN

3. Insert in LOG1:

ADC	$4000	Non-FORTRAN
ADC	$4000	FORTRAN

4. Insert in LOG2:

Label	Op	Address	Significance
	NUM	$FFFF	Non-FORTRAN
	NUM	$FFFF	FORTRAN

5. If the timer package is used, add the PHYSTB addresses to the diagnostic timer table.

	ADC	LP501	Non-FORTRAN
	ADC	LP501	FORTRAN

6. Construct a PHYSTB for the 1740-501 using the following instructions and sample PHYSTB as a guideline:

   a. Declare the driver entry points external.

   b. Word 0: select the priority level of the scheduler request so that it corresponds to the priority level selected in the appropriate interrupt trap area of LOCORE. Priority 10 is used in this example.

   c. Words 1,2,3: insert the addresses of the driver's initiator, continuator and error routine

   d. Word 7: insert the hardware equipment connect code. The example below uses equipment F, station 1

   e. Word 8: insert the request status word

   f. Word 19: insert the FORTRAN line printer logical unit number

Word	Label	Op	Address	Significance
		EXT	IN501, CN501, ER501, MAS501	
0	LP501	NUM	$120A	
1		ADC	IN501	Initiator
2		ADC	CN501	Continuator
3		ADC	ER501	Error routine
4-6		NUM	-1,0,0	
7		NUM	$781	Hardware address
8		NUM	$2934	Request status
9-12		NUM	0,0,0,0	
13		NUM	0	Driver length if mass memory
14		ADC	MAS501	Name associated with sector number
15-18		NUM	0,0,0,0	
19		ADC	FLP-LOG1A	FORTRAN logical unit

7. Install the driver under an *L statement.

8. If the driver is mass memory, replace word 13 of the PHYSTB with:

Word	Label	Op	Address	Significance
		NUM	$20E	Driver length if mass memory

2.8 1731-601 BUFFERED MAGNETIC TAPE DRIVER

2.8.1 INSTALLATION REQUIREMENTS

The following modules with corresponding memory requirements are necessary:

TAPDRB	471 words
FRWAB	247
FRWBB	245
RECVTB	148
TAPE	14
RWBAB	171
	1296 words of core memory

2.8.2 INSTALLATION PROCEDURES

Make the standard changes to LOCORE and SYSBUF which are listed in the introduction to part II, section 2. Also add space to AREAC of the SPACE driver for a buffer area which is three times the maximum buffer size [(MAXVAL)*3+1]. The following procedures outline only those items which are unique to the 1731 buffered magnetic tape driver.

1. Insert the following in the interrupt trap area of LOCORE using the desired interrupt line:

Label	Op	Address
LINEx	NUM	0
	RTJ-	($FE)
	NUM	11
	ADC	INT601

2. Insert into the LOG1A the first word label in the position corresponding to the interrupt line to be used:

Label	Op	Address
	EQU 2	Lx(*)
	ADC	TPPDR1
	ADC	TPPDR2

3. Insert in LOG1:

	ADC	0
	ADC	0

4. Insert in LOG2:

	NUM	$FFFF
	NUM	$FFFF

5. Insert a PHYSTB similar to the one below:

Word	Label	Op	Address	Significance
0	TPPDR1	NUM	$120B	
1		ADC	TAPDRB	
2		ADC	TAPCB	
3		ADC	TAPHB	
4-6		NUM	-1,0,0	
7		NUM	$1381	
8-12		NUM	$896,0,0,0,0	
13		NUM	0	Driver length if mass memory
14		ADC	MAS31	
15-16		NUM	0,$414	
17		ADC	TPPDR2	
18		NUM	192	Maximum record size
19		NUM	0	

6. Also insert the following entry points and externals:

```
Label       Op      Address
            ENT     TPPDR1
            EXT     TAPDRB,TAPCB,TAPHB
            EXT     MAS31
```

7. If the number of words requested is greater than the value in MAXVAL (word 18 of the PHYSTB), the number is truncated to the value in MAXVAL. No error message is output. MAXVAL is set to 192 in the standard system to allow COSY to run using records which are two sectors in length.

   Area 4 in SPACE must be increased. The algorithm is (MAXVAL)*3+1. For the standard system set area 4 to $241.

8. EPROC cannot be used as an interrupt response routine for buffered devices. A routine similar to the following must be inserted.

```
            ENT     INT601
INT601      LDQ     =XTPPDR1
            JMP*    (TPPDR1+2)
```

2.9 1731-601 UNBUFFERED MAGNETIC TAPE DRIVER

2.9.1 INSTALLATION REQUIREMENTS

The following modules with corresponding memory requirements, are necessary:

TAPEDR	495
RWBA	134
FRWA	213
FRWB	259
RECOVT	141
TAPE	14
	1286

2.9.2 INSTALLATION PROCEDURES

Make the standard changes listed in the introduction to part II, section 2 as well as following the procedures unique to the 1731-601.

1. Insert the following in the interrupt trap area of LOCORE using the desired interrupt line:

```
Label       Op      Address
LINEx       NUM     0
            RTJ-    ($FE)
            NUM     11
            ADC     EPROC
```

2. Insert into LOG1A the first word label of the PHYSTB in the position corresponding to the interrupt line used by the 601. For example, after EQU Lx(*), insert:

Label	Op	Address
	ADC	TPPDR1
	ADC	TPPDR2

3. Insert into LOG1:

	ADC	0

4. Insert into LOG2:

	NUM	$FFFF

5. Insert a PHYSTB similar to the following example:

Word	Label	Op	Address	Significance
		EXT	TAPEDR, TAPEC, TAPEH, MAS31	
0		NUM	$120B	
1	TPPDR1	ADC	TAPEDR	
2		ADC	TAPEC	
3		ADC	TAPEH	
4-6		NUM	-1,0,0	
7		NUM	$381	
8		NUM	$896	
9-12		NUM	0,0,0,0	
13		NUM	0	Driver length is mass memory
14		ADC	MAS31	
15-16		NUM	0,$414	
17		ADC	TPPDR2	

6. The magnetic tape drivers only allow formatted I/O. To have unformatted I/O for magnetic tape, add RWBA and delete *S,RWBA,7FFF from the install tape.

2.10 1732-608/609 MAGNETIC TAPE DRIVER

2.10.1 INSTALLATION REQUIREMENTS

A 1732-608/609 is necessary for a buffered magnetic tape driver; a 1732-609 is necessary for an unbuffered driver.

2.10.2 INSTALLATION PROCEDURES

Make the standard changes listed in the introduction to part II, section 2. Below are changes unique to the 1732-608/609.

1. For an unbuffered driver, insert an entry similar to the following into the appropriate interrupt trap area of LOCORE.

Label	Op	Address	Significance
LINEx	NUM	0	
	RTJ-	($FE)	
	NUM	11	Priority level
	ADC	EPROC	

2. For a buffered driver, an interrupt response routine must be inserted instead of using EPROC. Following is an example of an interrupt response routine for a buffered driver. This routine can be placed in SYSBUF.

```
INT608    LDQ     =XTAPDR1
          JMP*    (TAPDR1+2)
```

An interrupt trap entry, for the buffered 1732-608/609 using the interrupt response routine given above, is:

LINEx	NUM	0	
	RTJ-	($FE)	
	NUM	11	
	ADC	INT608	Interrupt response routine for buffered 608
	EXT	INT608	

3. Insert into the LOG1A table after EQU Lx(*):

Label	Op	Address
	ADC	TAPDR1

4. Insert in LOG1:

	ADC	0

5. Insert in LOG2:

	NUM	$FFFF

6. Insert in the diagnostic timer table:

	ADC	TAPDR1

7. Insert a PHYSTB similar to the following:

Word	Label	Op	Address	Significance
		EXT	TAPINT, TAPCON, TPHANG	
0	TAPDR1	NUM	$120B	Driver at level 11
1		ADC	TAPINT	Initiator
2		ADC	TAPCON	Continuator
3		ADC	TPHANG	I/O hang up
4		NUM	-1	Diagnostic clock
5		NUM	0	
6		NUM	0	
7		NUM	xxxx	$1281 for buffered driver $281 for unbuffered driver
8		NUM	xxxx	$A66 for 608 $A76 for 609
9		NUM	0	Status word 1
10		NUM	0	
11		NUM	0	

Word	Label	Op	Address	Significance
12		NUM	0	
13		NUM	0	Driver length
14		ADC	MAS32	
15		NUM	0	Temp. storage
16		NUM	xxxx	$4C0 for select unit 1
				$440 for select unit 0
				Bits
				15-11 Unused; set to zero
				10 Select; set to one
				9-7 Tape unit number (0-7)
				6 Assembly mode of data transfer; set to one
				5 Select 200 BPI; set to zero
				4 Select 556 BPI; set to zero
				3 Select 800 BPI; set to zero
				2 Binary (odd parity); set to zero
				1 BCD (even parity) 608 only; set to zero
				0 Not used
17		ADC	TAPDR2	Address of next PHYSTB
18		NUM	0	Temporary storage

8. DR1732 is coded as a macro skeleton. To parameterize the driver for a particular configuration, prepare a COSY control deck as follows:

Label	Op	Address
DR1732	DCK/	I=lu, H=lu
	DEL/	
TPDRGN	$P_1, P_2, P_3, P_4, P_5, P_6$	

The formal parameters P_1 through P_7 are defined as follows:

P_1 Defines the residency of the driver

CORE	For core resident
MASS	For mass memory resident. When MASS is specified the following additional deck card is needed.

TAPCOR DCK/ I=lu, H=lu

P_2 Defines the method of data transfer

BUF	For buffered transfers using a 1706 buffered data channel
UNBUF	For unbuffered transfers using the A/Q channels

P_3 Defines the type of tape drives which will be in the system: 608, 609, or BOTH

P_4 Defines the type of read/write requests to be processed

FORM	Only formatted requests are to be processed
REG	Only regular requests are to be processed
BOTH	Formatted and regular requests are to be processed

P_5 Defines whether error recovery for parity errors will be attempted

ERR	Recovery will be attempted
NOERR	The error bits are set; the request is completed

P_6 Defines the maximum tape record size for 608 units. If blank, 96 words are assumed. The standard binaries were made using 192. This was done to allow COSY to run records which are two sectors in length or 192 words.

9. To obtain source, decosy the necessary decks specified above.

10. Assemble the source of DR1732.

11. If the driver is to be in core resident, insert it in the source tape before the SPACE module as follows:

```
*L
DR1732      (binaries)
```

12. If the number of words requested is greater than the value specified in P_6 (step 8) the number is truncated. No error message is output. P_6 is set to 192 in the standard system to allow COSY to run using records which are two sectors in length.

    Area 4 in SPACE must be increased. The algorithm is $(P_6)*4/3+2$. For the standard system set area 4 to $102.

2.11 1777 PAPER TAPE STATION

2.11.1 GENERAL 1777 PAPER TAPE STATION INFORMATION

Description

The 1777 paper tape station driver drives either:

the 1777 paper tape station or,

the 1721/1722 paper tape reader or,

the 1723/1724 paper tape punch or,

both the 1721/1722 paper tape reader and the 1723/1724 paper tape punch.

The 1777 is composed of two drivers: the 1777 paper tape station punch and the 1777 paper tape station reader.

Limitations

1704: The 1777 paper tape station driver is on equipment 1 and interrupt line 1 if it is used to drive the 1721/1722 paper tape reader and/or the 1723/1724 paper tape punch. However, because of the low speed common synchronizer, the 1777 paper tape station driver cannot be on equipment 1 if it drives the 1777.

1774 S. C.: When the 1777 paper tape station is used, there are no unique interrupt line and equipment number restrictions.

Installation Procedures

The 1777 paper tape station driver is either mass memory or core resident, depending on equate MM.

Core Resident Installation:

1. Equate MM to 0
2. Load the 1777 paper tape station reader and punch drivers under the *L statement.
3. Load STCK under the *L DRCORE. STCK (status check) is the program containing all common routines for the 1777 station reader and punch drivers. It is used when the 1777 paper tape station reader and punch drivers are core resident. When the 1777 paper tape station reader and punch drivers are mass memory, STCK is included in each program since program length is not important.

Mass Memory Installation:

If installing on mass memory, see part II, section 2.15.

2.11.2 1777 PAPER TAPE STATION READER DRIVER

Description

The 1777 paper tape station reader driver allows data input from the paper tape reader to core memory. The reader driver is re-entrant so that it can handle multiple readers.

Installation Requirements

Mass Memory:

Driver	256
Logical unit tables	3
Diagnostic timer table	1
Physical equipment table	22
	282 words of mass memory

Core Memory:

Driver	227
Logical unit tables	3
Diagnostic timer table	1
Physical equipment table	22
	253 words of core memory

Installation Procedures

The following installation procedures are unique to the 1777 paper tape station reader driver.

1. The following example is a four word interrupt trap entry associated with the low speed I/O common synchronizer. The example is assigned to line 1, priority level 10.

```
Label     Op      Address
LINE1     NUM     0
          RTJ-    ($FE)
          NUM     10
          ADC     EPROC
```

2. Insert in LOG1A:

Label	Op	Address
	ADC	PPTRDR

3. Insert in LOG1:

	ADC	0

4. Insert in LOG2:

	NUM	$FFFF

5. Add the PHYSTB to the system tables and parameters using the following coding. In this example the priority level is 10, the equipment type is 1, the equipment class is 4.

Word	Label	Op	Address	Comments
		EXT	TR1777	
		EXT	PREADI, PTREAD, PTRERR	
0	PPTRDR	NUM	$120A	
1		ADC	PREADI	
2		ADC	PTREAD	
3		ADC	PTRERR	
4-6		NUM	-1, 0, 0	
7		NUM	$A1	
8		NUM	$2012	
9-12		NUM	0, 0, 0, 0	
13		NUM	0	Driver length if mass memory
14		ADC	TR1777	Name associated with sector number
15-21		BZS	(7)	

6. Add the following entry to the diagnostic timer table (DGNTAB) if time-out surveillance on the reader operation is desired:

	ADC	PPTRDR

7. Modify MASKT according to instructions in part II, section 1.2.3.

8. If the driver is mass memory, replace word 13 of the PHYSTB as follows:

	NUM	$100

2.11.3 1777 PAPER TAPE STATION PUNCH DRIVER

Description

The 1777 paper tape station punch driver allows data output from core memory to the paper tape punch. The driver punches eight level tape only. The punch driver is re-entrant so that it can handle multiple punches.

Installation Requirements

Mass Memory:

Driver	240
Logical unit tables	3
Diagnostic timer table	1
Physical equipment table	22
	265 words of mass memory

Core Memory:

Driver	165
Logical unit tables	3
Diagnostic timer table	1
Physical equipment table	22
	191 words of core memory

Installation Procedures

The following installation procedures are unique to the 1777 paper tape station punch driver.

1. The following example is a four word interrupt entry associated with the low speed I/O common synchronizer. The example is assigned to line 1, priority level 10.

```
Label     Op      Address
LINE1     NUM     0
          RTJ-    ($FE)
          NUM     10
          ADC     EPROC
```

2. Into LOG1A enter:

Label	Op	Address
	ADC	PPTPCH

3. Into LOG1 enter:

	ADC	0

4. Into LOG2 enter:

	NUM	$FFFF

5. Enter as an external:

	EXT	TP1777, PUNCHI, PUNCDR, PUNERR

6. Insert the following PHYSTB. The driver priority level is 10, the equipment class is 4, the equipment type is 2.

Word	Label	Op	Address	Comments
		ADC	TP1777	
0	PPTPCH	NUM	$120A	
1		ADC	PUNCHI	
2		ADC	PUNCDR	
3		ADC	PUNERR	
4-6		NUM	-1, 0, 0	
7		ADC	$C1	
8		ADC	$2024	
9-12		NUM	0, 0, 0, 0	
13		NUM	0	Driver length if mass memory
14		ADC	TP1777	Name associated with sector number
15-21		BZS	(7)	

7. If time-out surveillance over punch operations is desired, insert the following entry into the diagnostic timer table:

Label	Op	Address
	ADC	PPTPCH

8. Modify MASKT according to instructions in part II, section 1.2.3.

9. Validation Option Procedures: If the validation check and repunch is required, equate VALERR to 1 in the driver source, and assemble the driver. This is a hardware feature which is available only on the 1777 paper tape station.

```
EQU     VALERR(1)
```

10. If the driver is mass memory replace word 13 of the PHYSTB as follows:

```
NUM     $112
```

2.12 1711/1712/1713 TELETYPEWRITER DRIVER

2.12.1 DESCRIPTION

The 1711/1712/1713 teletypewriter driver executes under the MSOS operating system to provide the capability for data input/output between core memory and the teletypewriter keyboard. The teletypewriter connects directly to the 1704 computer and is part of the low-speed I/O common synchronizer package.

The 1711/1712/1713 driver processes requests made by user programs for data transfer between core memory and the teletypewriter. The requests are READ, FREAD, WRITE, and FWRITE.

2.12.2 INSTALLATION REQUIREMENTS

Core Memory

Driver	319 words
Logical unit tables	3 words
Diagnostic timer table	1 word
Physical equipment table	16 words
	339 words of core memory

Mass Memory

None

2.12.3 INSTALLATION PROCEDURES

The equipment code is preset to one for all low-speed I/O common synchronizer devices.

1. The following four-word interrupt entry must be in the interrupt trap area of the LOCORE program. It is associated with the low-speed I/O common synchronizer package and is assigned to interrupt LINE1:

```
Label    Op      Address
LINE1    NUM     0
         RTJ-    ($FE)
         NUM     10
         ADC     EPROC
```

2. Enter the following into LOG1A:

```
ADC     TELPTR
```

3. Enter the following into LOG1:

```
ADC     0
```

4. Enter the following into LOG2:

```
ADC     $FFFF
```

5. Declare the following entry point:

```
ENT     TELPTR
```

6. Declare the following external:

```
EXT     TYPEI,TYPEDR,TYPERR
```

7. In forming the PHYSTB for the 1711/1712/1713 teletypewriter driver, use the following information:

driver priority level	10
equipment type	0
equipment class	6

Add the PHYSTB to the system tables and parameters (SYSBUF) using the following coding:

Word	Label	Op	Address
0	TELPTR	NUM	$120A
1		ADC	TYPEI
2		ADC	TYPEDR
3		ADC	TYPERR
4-6		NUM	-1,0,0
7-8		ADC	$91,$3006
9-15		BZS	(7)

8. If time-out surveillance over teletypewriter operation is desired, enter into the diagnostic timer table in SYSBUF the following entry:

```
ADC     TELPTR
```

9. Modify MASKT according to instructions in part II, section 1.2.3.

2.13 1713 TELETYPEWRITER READER/PUNCH DRIVER

2.13.1 DESCRIPTION

The 1713 teletypewriter reader/punch driver provides either keyboard, or paper tape and printer, or paper tape output. The reader and punch modules reside on mass memory. The keyboard module must be core resident. It processes requests made by user programs between core memory and the teletypewriter.

2.13.2 INSTALLATION REQUIREMENTS

Core Memory

MASDRV and buffer (equated to LNGTH)	142+buffer size
System tables and parameters	12
Physical equipment tables	60
Common continuator (S13CON)	11
Diagnostic timer table	3
Keyboard printer	379
	607 + buffer size

Mass Memory

S13002 reader	274
S13003 punch	272
	546

2.13.3 INSTALLATION PROCEDURES

The following procedures are unique to the 1713 teletypewriter reader/punch driver.

1. The 1713 is part of the low-speed package which is loaded into the computer on interrupt line 1. Only the keyboard device table address must be included with the other device addresses using line 1. The 1713 reader and punch may be independent of line 1, since the continuator of the modules determines which module is active. Therefore, the reader and punch may be assigned any logical unit numbers. The 1713 reader and punch should not be assigned to any other interrupt line. Refer to part II, section 1.2.2 for information on interrupt line assignment.

Label	Op	Address	Comments
LINE1	NUM	0	
	RTJ-	($FE)	
	NUM	10	
	ADC	EPROC	

2. Insert the following into the LOG1A table:

ADC	S13KBD	Entry in keyboard PHYSTB
ADC	S13RDR	Entry in reader PHYSTB
ADC	S13PCH	Entry in punch PHYSTB

3. Insert the following into LOG1:

```
ADC     0
ADC     0
ADC     0
```

4. Insert the following into LOG2:

```
NUM     $FFFF
NUM     $FFFF
NUM     $FFFF
```

5. Insert the following into SECPRO for each module:

```
NUM     $7FFF
```

6. If the timer package is to be used, add the device table addresses to the diagnostic timer table.

```
ADC     S13KBD
ADC     S13RDR
ADC     S13PCH
```

7. Declare the following:

```
ENT     S13BZY,S13MOD
ENT     S13KBD,S13RDR,S13PCH
ENT     S13CON
EXT     S13KI,S13KC,S13KER
EXT     MASDRV,MI,MASHNG
EXT     M1713R,M1713P
```

8. Insert the following physical equipment tables:

Word	Label	Op	Address	Comments
	S13BZY	NUM	0	
	S13MOD	NUM	0	
*KEYBOARD PHYSTB				
		ADC	S13KC	Keyboard continuator
0	S13KBD	NUM	$120A	
1		ADC	S13KI	
2		ADC	S13CON	
3		ADC	S13KER	
4-8		NUM	-1,0,0,$91,$3266	
9-12		NUM	0,0,0,0	
13		NUM	0	
14		NUM	0	Must be zero
15-16		NUM	0,0	
*READER PHYSTB				
	S13RC	ADC	0	
0	S13RDR	NUM	$120A	
1		ADC	MASDRV	
2		ADC	S13CON	
3		ADC	0	
4-8		NUM	-1,0,0,$91,$2282	
9-12		NUM	0,0,0,0	
13		NUM	$112	Driver length
14		NUM	M1713R	Name associated with sector number
	*			
15-18		NUM	0,0,0,0	

Word	Label	Op	Address	Comments
*PUNCH PHYSTB				
	S13PC	ADC	0	
0	S13CON	NUM	$120A	
1		ADC	MASDRV	
2		ADC	S13CON	
3		ADC	MASHNG	
4-8		NUM	-1,0,0,$91,$2274	
9-12		NUM	0,0,0,0	
13		NUM	$11A	Driver length
14		ADC	M1713P	Name associated with sector number
	*			
15-17		NUM	0,0,0	
*COMMON CONTINUATOR				
1	S13CON	LDQ*	S13BZY	
2		SQN	SA	
3		ENQ	KBDLU	
4-5	SA	LDQ	LOG1A,Q	
6		TRQ	A	
7		INQ	-1	
8		LDQ-	(ZERO),Q	
9		STQ-	I	
10		TRA	Q	
11		JMP-	(ZERO),I	

9. Equate the logical unit of the keyboard to KBDLU as:

```
        EQU     KBDLU(4)
```

10. Names which are associated with reader and punch sectors are:

reader module	*S, M1713R, S
punch module	*S, M1713P, S

11. Load the punch and reader modules under separate *M statements with *S statements immediately after the respective relocatable binary.

```
*M        1713 READER
S13002
*S, M1713R, S
*M        1713 PUNCH
S13003
*S, M1713P, S
```

12. Install the keyboard module, S13001, and MASDRV as core resident modules under an *L statement.

13. Modify MASKT as in part II, section 1.2.3.

2.14 1572/1573 TIMER

Install the timer (TIMINT) routine at system initialization time.

1. Rebuild the installation tape to include the TIMINT routine. Before assembling the system tables, initialize the following timer external parameters which define system variables and are necessary for timer operation:

   TIMCPS — Determines timer operating frequency. If, for example, TIMCPS is equated to 60 (as in the example below) the timer interrupts every 60 seconds.

   TIMACK — Contains equipment and station used to acknowledge timer interrupt. Modify this parameter if the equipment number for the 1750 Data and Control Terminal is also modified.

   NSR — Establishes upper limit on the number of timer completion addresses scheduled for each timer interrupt. Excess addresses are handled on the next interrupt.

2. Insert the following coding sequence in the System Tables for the 1572/1573

Label	Op	Address	Comments
	ENT	TODLVL	Time of day timer req. with level
	ENT	TIMCPS	Timer cycles per second
	ENT	TIMEC	Timer cycles per 1 sec -1
	ENT	TIMACK	Timer achnowledge code
	ENT	NSCHED	Max. num. of comp adrs per int
TIMCPS	EQU	TIMCPS (60)	
TIMEC	EQU	TIMEC (TIMCPS/10-1)	
TIMACK	EQU	TIMACK ($401)	
NSR	EQU	NSR(5)	
NSCHED	ADC	NSR	
TODLVL	EQU	TODLVL ($1,006)	

3. Initialize interrupt line x in LOCORE so that it will accommodate the timer interrupt. Insert the following coding in the interrupt trap area of LOCORE:

LINEx	NUM	0	
	RTJ-	($FE)	Common interrupt handler
	NUM	13	Priority level of timer INT
	ADC	TIMINT	1572/1573 timer interrupt processor

4. For the 1572 timer add the following

Label	Op	Address	Comments
	ENT	TIVALU	
TIVALU	EQU	TIVALU (xxxx)	xxxx is a value from 1 to 4000

5. If the 1572 timer is used, replace TIMACK with the following:

TIMACK	EQU	TIMACK($0xx3)	Bits 7-10 contain the equipment code

2.15 MASS MEMORY DRIVERS

If a driver is to be placed on mass memory, use the following instructions. For an example of an installation tape see part III, section 7.1.3.

1. Assemble the drivers that are to reside on mass memory, setting the equate to 1

Label	Op	Address
	EQU	MM(1)

2. For each mass memory resident driver, insert the length of the driver into word 13 of the driver PHYSTB. Following is a list of these values.

Driver	Length
Card Reader	
1726-405 unbuffered	$173
1726-405 buffered	$194
1728-430	$37A
1729-2	$1CC
Line Printer	
1740-501	$20E
Magnetic Tape	
1731-601 buffered	$51F
1731-601 unbuffered	$4B8
1732-608/609 buffered	$3EC
1732-608/609	$3CF

Driver	Length
Paper Tape Station	
1777 reader	$FF
1777 punch	$F8
Teletypewriter	
1713 reader	$112
1713 punch	$110

3. Assemble the system tables and parameters program.

4. There are two options for the mass memory driver control program:

   MASDRV for single buffered. MASDRV requires a buffer length the size of the largest driver residing on mass memory.

   DBLDRV for double buffered. DBLDRV requires a buffer length the size of the two largest drivers residing on mass memory.

   Several assembly options in MASDRV and DBLDRV must be chosen to fit the configuration. These options are the same in MASDRV and DBLDRV.

   a. Leave the priority level at which the control program runs at 10 for the standard system. No driver residing on mass memory can run below this priority level.

Label	Op	Address	Comments
	EQU	CMPRL(10)	

   b. Equate the number of drivers residing on mass memory:

```
        EQU     NMASDR( )
```

      The 1777 and the 1713 must each be counted as two drivers.

   c. Equate the length of the largest driver for MASDRV or the length of the two largest drivers for DBLDRV:

```
        EQU     LNGTH( )
```

   d. Equate the following values according to which magnetic tape driver is mass memory resident. Only one magnetic tape driver may be mass memory resident. If none of the magnetic tape drivers are mass memory resident, set all of the following equates to zero.

      Equate MT to one if any magnetic tape driver is mass memory resident:

```
        EQU     MT(1)
```

Equate one of the following to one, according to the magnetic tape driver selected:

Label	Op	Address	Comment
	EQU	D1731( )	1731/1732 unbuffered (TAPEDR)
	EQU	B1731( )	1731/1732 buffered (TAPDRB)
	EQU	D1732( )	1732-608/609 (DR1732)

If any of the following drivers are selected to reside on mass memory, set the respective equate to a one. If any of the drivers are core resident or are not used, set the equate to zero:

EQU	D1726( )	1726-405 card reader
EQU	D1728( )	1728-430 card reader
EQU	D1729( )	1729-2 card reader
EQU	D1740( )	1740-501 line printer
EQU	D1777R( )	1777 paper tape reader
EQU	D1777P( )	1777 paper tape punch
EQU	D1713( )	1713 reader and punch

5. Install MASDRV or DBLDRV as a core resident module under an *L statement and before any of the mass memory resident drivers.

6. Install the mass memory drivers under a separate *M statement with the *S,name,S statement placed immediately after the relocatable binary of the mass memory resident driver. The names referred to in the *S statement are associated with the sector number of the drivers when they are mass memory resident. These names, which are listed below, are declared external in SYSBUF.

Card reader

1726-405	MAS405
1728-430	MAS28
1729-2	MAS292

Line printer

1740-501	MAS501

Magnetic tape

1731	MAS31
1732	MAS32

Paper tape

1777 reader	TR1777
1777 punch	TP1777

Teletypewriter

1713 reader	M1713R
1713 punch	M1713P

Example:

```
*M
    PRT40    address
*S,MAS501,S
*M
    CR405    address
*S,MAS405,S
*M
    DR1732   address
*S,MAS32,S
```

7. Delete the respective *S,name,7FFF from the standard release tape.

OTHER MODIFICATIONS 3

Section 3 defines the following possible modifications:

3.1 BUILDING AN INITIALIZER

3.1.1 AVAILABLE MODULES

CONTRL†	Control module
LIB†	Library generation module
IDRIV†	Input control module (input device driver)
MDRIV†	Mass storage driver control module
CDRIV†	Comment control module (comment device driver)
ILOAD†	Resident loader
I1†	Pre-resident load initialization
I2†	Post-resident load initialization
MSDISK†††	Pre-resident initialization 853/854 disk driver
MSDRUM†††	Pre-resident initialization 1751 drum driver
I2DISK†††	Post-resident initialization 853/854 disk driver
I2DRUM†††	Post-resident initialization 1751 drum driver
MTIDRV††	601 magnetic tape driver

† Required modules
†† Optional according to configuration
††† Normally either the disk or drum drivers are used, not both

PTIDRV††	1721/1722 paper tape reader driver
LPRINT††	1742 line printer driver
CDIDRV††	1728-430/1729-2/1726-405 card reader drivers

3.1.2 PROCEDURES FOR GENERATING AN INITIALIZER

1. Obtain all necessary and optional modules from MSOS COSY tape, and assemble them.
2. Use the relocatable binaries received as input to LIBEDT.
3. Assign input to the logical unit containing the binaries.
4. Type LIBEDT

   Press RETURN
5. Type *K, Ilu

   Press RETURN
6. Type *P

   Press RETURN
7. All input is read

   Message LUluFAILED 02
8. Type CU

   Press RETURN
9. At this time, the absolutized binary is punched on the paper tape. The following is printed on the list device as unlinked. This is to provide easy linking of user modules when required.

   I3

   I4
10. For tape format, see part III, section 7.1. Record 1 of the initializer tape is the absolutized checksum loader. Record 2 of the initializer is the absolutized binary programs. Therefore, when the binaries have been absolutized, insert them after the checksum loader which has been assembled and absolutized.

ABSOLUTIZED	G	ABSOLUTIZED BINARY
CHECKSUM	A	
LOADER	P	INITIALIZER

†† Optional according to configuration

3.2 MANUAL INPUT FOR PROCESS PROGRAM (MIPRO)

The manual input to the process program (MIPRO) is part of the operating system. If the input entered after a manual interrupt does not begin with an asterisk (* indicates a job processor control statement), the routine is scheduled by the manual interrupt processor (MINT) at level 3. The Q register is set to the address of the ASCII input buffer on entry to MIPRO.

If the MIPRO program is not included in the operating system at initialization time, the manual interrupt processor rejects input following a manual interrupt which does not begin with an asterisk. A J05 error message is printed.

The version of MIPRO which is supplied checks the input buffer for either DB or DX. All other inputs are rejected and the message ER is printed. If the input begins with DB, the program with the system directory entry name ODEBUG (On-Line Debug Package) is scheduled at level 3. If the input begins with DX, a flag (CHRSFG) in SYSBUF is set for the ODEBUG routine. When this flag is set, ODEBUG terminates and releases its core.

MIPRO must terminate by clearing the flag word MIB in the manual interrupt processor and then returning to the dispatcher.

MIPRO usually resides on mass storage as part of the system library, but it may be made core resident. Each user may add his own control statements to MIPRO to manually control the process.

To add a user request to MIPRO:

1. For the entry point of the request processor module, add the following with xxxxxx as the entry point:

```
Label      Op      Address

           EXT     xxxxxx
```

2. Add to the end of the COD1 table:

```
           ALF     1,xx
```

   xx are the same control characters used to call the user program.

3. Add the following entry in the same numeric position as it is in the COD1 table:

```
           JMP*    GETIND
```

4. Add the following to the INDEX table with xxxxxx as the entry point of the request processor module:

```
           ADC     (xxxxxx)
```

3.3 USER REQUEST MODULES

3.3.1 PROCEDURES

The 1700 operating system allows 30 request processors in the standard release. The first 20 of these (T1-T20) are reserved for the operating system. The last 10 may be designated by the user (T21-T30).

Add a request processor to the system by supplying a processing program for the request and assigning an entry point name of T21 to T30 to it. Include this program in the System Load as a core resident entry and remove the *S which links it to $7FFF. The request processor to be added to the system must adhere to the following restrictions:

The entry point name must be one from T21 to T30

Enter the request processor program by entering the location of the parameter list into the A register

The request processor must exit with a jump to the request exit entry point REQXT

3.3.2 CALLING SEQUENCE

A typical calling sequence to the request module is:

Label	Op	Address	Significance
	RTJ-	($F4)	Go to monitor
	NUM	$hhhh	Request code: bits 14-09 contain the processor request number which is contained in the entry point name (T21-T30).

3.4 RE-ENTRANT FORTRAN LIBRARY PACKAGE

3.4.1 PREPARATION

Priority Level

If the re-entrant FORTRAN library package is to be used in the system and run at more than one priority level, replace the NDISP and SCHEDU modules with the RDISP module. If the FORTRAN library package is only to be run at one level, RDISP is not necessary; then remove FMASK and FLIST from SYSBUF.

FMASK

FMASK is a location which indicates the software priority levels requiring the saving of the temporary area used by the FORTRAN routines. Do not assign these levels to interrupt lines, since the interrupt handler does not save the FORTRAN data. Set to one each bit position in FMASK that corresponds to each level using FORTRAN. If too many levels are allowed to run FORTRAN programs, the overhead for the low-priority programs may be unnecessarily high.

Example:

Label	Op	Address
FMASK	NUM	$008C

This allows FORTRAN at levels 2, 3, and 7.

Levels 0 and 1 are reserved for unprotected programs and do not interrupt higher priority levels using FORTRAN. Therefore, the mask is not set for levels 0 and 1. The mask usually is not set for 6 because level 6 is usually used for the invalid interrupt processor.

Table FLIST is the table of entry point locations in the FORTRAN library which must be saved to allow re-entrant use of the library. The symbolic names must also be declared as externals (EXT) and must appear as entry names (ENT) in the library subroutines.

```
FLIST   ADC     FEND-*-1
        ADC     Q8QF21,Q8Q12F,Q8QF2F,RETAD,QSAVE
        ADC     Q8PREP,Q8PKUP,Q8AB,ABS,IFALT,Q8SG
        ADC     SIGN,Q8QFIX,Q8FX,Q8QFLT,Q8FLOT
        ADC     IFIX,FLOAT,EXP,SQRT,ALOG,TANH
        ADC     SIN,COS,ATAN,FLOT,ARGU0
        EQU     FEND(*)
        EXT     Q8QF21,Q8Q12F,Q8QF2F,RETAD,QSAVE
        EXT     Q8PREP,Q8PKUP,Q8AB,ABS,IFALT,Q8SG
        EXT     SIGN,Q8QFIX,Q8FX,Q8QFLT,Q8FLOT
        EXT     IFIX,FLOAT,EXP,SQRT,ALOG,TANH
        EXT     SIN,COS,ATAN,FLOT,ARGU0
```

Q8STP

Also add the entry Q8STP to SYSBUF for use with the re-entrant library package:

Label	Op	Address
	ENT	Q8STP
Q8STP	NOP	0
	JMP-	($EA)

3.4.2 INSTALLATION PROCEDURES

1. Assemble RDISP and obtain a relocatable binary
2. Put the changes mentioned above under preparation into SYSBUF; assemble SYSBUF; obtain a relocatable binary
3. Assemble and obtain binaries of the desired FORTRAN library routines

Re-Entrant Encode/Decode Routines

IOCODE
INITAL
RSTORE
IGETCH
IPACK
UPDATE
DECPL
INTGR
SPACEX
HOLRTH
DCHX
HXASC
AFRMOT
RFRMOT
AFRMIN
RFRMIN
ASCHX
HXDC
FLOTIN
FOUT
EOUT
EWRITE
FORMTR

Q8QFI
Q8QFX
AFORM
RFORM
HEXASC
HEXDEC
ASCII
DECHEX
FLOATG
INITL1
FORMTR
Q8QFI
Q8QFL
Q8QFX

FORTRAN I/O Routine

Q88IO

Monitor Interface Routine

FORTRA

Arithmetic Routines

Q8EXPR
Q8PRMS
Q8ABR
IFALTR
SIGNR
FXFLR
EXFLR
EXPRGR
SQRTFR
LNPRGR
TANHR
SNCSR
ARCPGR
FLOATR

4. Replace SYSBUF with the revised version of SYSBUF on the installation tape.
5. Also replace NDISP and SCHEDU with RDISP on the installation tape.
6. Insert the desired modules of the FORTRAN library package (listed under step 3) into the installation tape under an *L statement.
7. Install the MSOS 3.0 system as it would normally be installed in part I, section 1.3.

3.5 NON-RE-ENTRANT FORTRAN

3.5.1 REQUIREMENTS

None

3.5.2 INSTALLATION PROCEDURES

Non-re-entrant FORTRAN can be installed at the same time as MSOS 3.0 using an initializer *F statement.

1. Assemble and obtain relocatable binaries of the desired routines.

   Encode/Decode

   IOCODE
   IGETCH
   IPACK
   UPDATN
   DECPL
   INTGR
   SPACEN
   HOLRTH
   DCHX
   HXASC
   AFRMOT
   RFRMOT
   AFRMIN
   RFRMIN
   ASCII
   HXDC
   FLOTIN
   FOUT
   EOUT
   EWRITE

AFORM
RFORM
HEXASC
HEXDEC
DECHEX
FLOATG
FORMTR
INITL1
ASCHX
PSEUDO
Q8QFI
Q8QFL
Q8QFX
Q8QIO
Q8QGTX

Monitor Interface

FORTN

Arithmetic Routines

Q8EXPN
Q8PRMN
Q8AB
IFALT
SIGN
FXFL
EXPPRG
SQRTF
LNUPRG
TANH
SINCOS
ARCTPG
FLOAT

2. Update the MSOS 3.0 installation tape listed in part III, section 7.1 by inserting the desired relocatable binary programs after the *F statement on the installation tape.

3. When the system is installed, the program library will also be generated.

3.6 OUTPUT MESSAGE BUFFERING PACKAGE

3.6.1 REQUIREMENTS

Reserve buffer area in core or in mass memory for exclusive use of the buffer package at system initialization time. Reserve at least three times the maximum record size.

The user may reserve the last 1000 sectors of the disk for the software buffering package. SECTOR is set to 2FFF during initialization, and the remaining sectors are used.

For core buffering, put a BSS block in SYSBUF.

For mass memory buffers, a *M, hhhh, s initializer statement may also be used.

A word addressable disk or drum driver is necessary for this version.

Replace the normal version of SYSBUF with one which defines the physical device tables for the software buffered devices.

Include BUFFER as a core resident program.

3.6.2 INSTALLATION PROCEDURES

1. The output message buffering package is added to the 1700 Operating System in the same way as is a driver. Insert a buffer table and a character buffer area for each buffer input logical unit by using the BUFFER macro. The BUFFER macro generates a physical device table for each buffered device.

   BUFFER f, l, h, lu, rp, n

   f start address of buffer (LSB)

   l end of buffer address plus 1

   h most significant bits of mass memory buffer word address; to be blank for core buffer

   lu logical unit for actual output

   pr request priority for buffer output on this logical unit

   n character buffer size for actual output

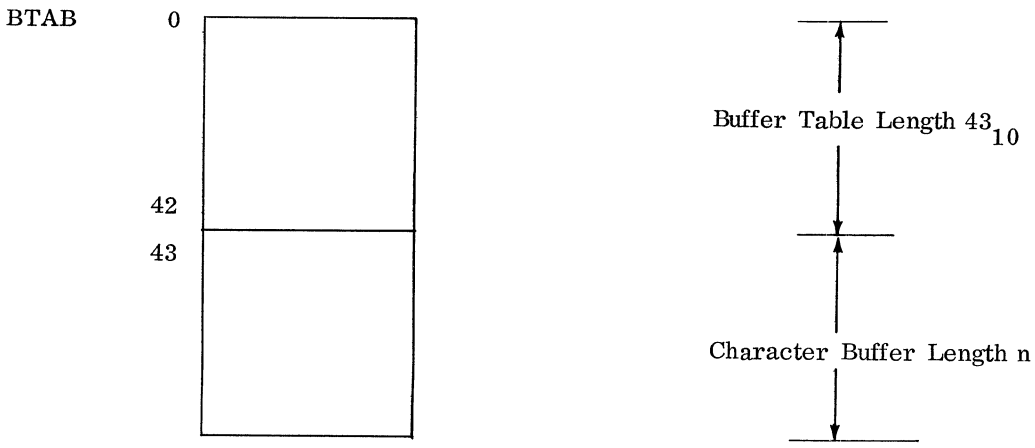
   When using the BUFFER macro, define the internal symbols as in the following example:

Label	Op	Address
	EQU	BFLEVL(10)
	EQU	BFMMLU($8C2)

BFLEVL is the priority level of the buffer package

BFMMLU is the mass memory device logical unit

The BUFFER macro generates the following:



n is the length of the character buffer specified in the macro call

BTAB is the address of the buffer table that must be put in the LOG1A logical unit table

The first 13 words of the table correspond to the standard 13 words required for all the physical equipment tables for all devices. The additional parameters define buffering parameters, the available core or mass memory area, and the character buffer size. The character buffer follows the last word of the buffer table.

2. Add a buffer table address to LOG1A:

Label	Op	Address
	ADC	BTAB1

3. Add to LOG1:

	ADC	0

4. Add to LOG2:

	ADC	$FFFF

5. Repeat steps 1 through 4 for each buffer input logical unit.

6. Insert the following macro in SYSBUF:

```
Label     Op      Address                          Comments
BUFFER    MAC     F,L,H,LU,RP,N
          LOC     A
          EXT     BUFDRI,BUFDRC
          EXT     BWRITC,BREADC,BOUTPC

          ADC     $1200+BFLEVL                     0
          ADC     BUFDRI,BUFDRC,BUFDRC
                                                   1-3
          NUM     -0,0,0,0,$28A4                   4-8
          NUM     0,0,0,0,0                        9-13
          ADC     ≠F≠,≠L≠,≠F≠,≠L≠                  14-17
          ADC     $04F0+BFLEVL                     18
          ADC     BWRITC,0,BFMMLU                  19-21
          ADC     0,0                              22-23
          IFC     ≠H≠,NE,
          ADC     ≠H≠                              24      M.M. BFR
          EIF
          IFC     ≠H≠,EQ,
          NUM     $8000                            24      CORE BFR
          EIF
          ADC     ≠F≠,0                            25-26
          ADC     BFLEVL*16+BFLEVL+$0200
                                                   27
          ADC     BREADC,0,BFMMLU                  28-30
          ADC     0,≠A≠,≠H≠,≠F≠,0                  31-35
          VFD     N8/$0C,X4/≠RP≠,X4/BFLEVL
                                                   36
          ADC     BOUTPC,0,≠LU≠                    37-39
          ADC     0,≠A≠,≠N≠                        40-42
          BZS     ≠A≠(≠N≠)                         CHAR BFR
          EMC
```

7. The following devices can be software buffered:

   1711/1712/1713 teletypewriter

   1728-430 card punch

   1777 paper tape station punch

   1740-501 line printer

8. Using the macro above, construct a PHYSTB for each buffered device desired as follows:

   Set the BUFFER macro up using sector 3000 and logical unit 4, the teletypewriter. The macro is as follows:

Label	Op	Address
	BUFFER	0,7FFF,$24,4,3,26

9. Assemble the BUFFER module and obtain a relocatable paper tape.

10. Insert BUFFER and replace the existing SYSBUF with a revised version of SYSBUF on the installation tape as core resident programs.

11. Install the system as it would normally be installed.

OUTPUT BUFFERING PACKAGE PHYSTB GENERATED BY BUFFERED MACRO

0	$1200+LV		ELVL	STANDARD PHYSTB
1	BUF DRI	Initiator Entry	EDIN	
2	BUFDRC	Continuator Entry	EDCN	
3	BUFDRC	Diagnostic Entry	EDPGM	
4	-1	Diagnostic Clock	EDCLK	
5	0	Logical Unit Assigned	ELU	
6	0	Request Address	EPTR	
7	0	Hardware Address	EWES	
8	$A4	Type Code	EREQST	
9	0	Status Word 1	ESTAT1	
10	0	Start Core Address	ECCOR	
11	0	End Core Address +1	ELSTWD	
12	0	Status Word 2	ESTAT2	
13	0	Number of Attempts	TIMER	
14	F	Buffer Start	LOCB	
15	L	Buffer End +1	ENDB	
16	F	Temporary Buffer Start	FIRST	
17	L	Temporary Buffer End +1	LAST	
18	$04F 0+LV	Mass Memory WRITE	DPL0	
19	BWRITC	Completion Address	1	STANDARD PHYSTB
20	0	Thread	2	
21	BFMMLU	Logical Unit	3	
22	0	Length	DLEG	
23	0	Core Address	DART	
24	H	MSB of Mass Memory Address	DTRACK	
25	F	Buffer Store Pointer	STOR	

OUTPUT BUFFERING PACKAGE PHYSTB GENERATED BY BUFFERED MACRO (contd)

26	0	Control Word	CONTRL	
27	$200+16*LV+LV	Mass Memory READ	DOUT0	
28	BREADC	Completion Address	1	STANDARD PHYSTB
29	0	Thread	2	
30	BFMMLU	Logical Unit	3	
31	0	Length	OUTLNG	
32	CHBUFF	Core Address	DADR	
33	H	MSB of Mass Memory Address	OUTTK	
34	F	Buffer Read Point	READ	
35	0	Control Word	SKELNG	
36	$C00+16*RP+LV	Character Output FWRITE	OUTP0	
37	BOUTPC	Completion Address	1	STANDARD PHYSTB
38	0	Thread	2	
39	LU	Output Logical Unit	3	
40	0	Length	4	
41	CHBUFF	Address of Character Buffer	ACHAR	
42	N	Length of Character Buffer	LCHAR	

CONVENTIONS 1

1. Terminate each teletype input by pressing RETURN.
2. To erase a teletype line:

   Type RUB OUT, LINE FEED

   Press RETURN

3. After mounting a paper tape on the paper tape reader, press READY MASTER CLR on the reader.
4. If the teletype BREAK light is on, press BRK RLS on the teletypewriter before attempting to type.
5. Core locations are base 16; lengths are base ten.
6. When using the 1713 teletypewriter, it must be in K mode.

UNIT ASSIGNMENTS 2

2.1 LOGICAL UNIT, EQUIPMENT, AND INTERRUPT LINE

The released MSOS 3.0 system configuration is:

Device	Logical unit	Interrupt line	Equipment
Core allocator	1	-	-
Card equipment			
1726-405	13	6	4
1728-430	10	10	8
1729-2	11	11	C
Drum			
1738-853 unit 0	8	4	3
Line Printer			
1740-501	9	5	F
501 FORTRAN	12	5	F
Magnetic tape equipment			
1731-601 unit 0	6	3	7
1731-601 unit 1	7	3	7
1732-608 unit 2	14	7	5
1732-608 unit 3	15	7	5
1732-609 unit 4	16	7	5
1732-609 unit 5	17	7	5
Paper tape equipment			
1777 reader	2	1	1
1777 punch	3	1	1
Teletypewriter			
1713 keyboard	4	1	1
1713 punch	19	1	1
1713 reader	18	1	1
Dummy	5	-	-

2.2 INITIALIZER LOGICAL UNIT AND EQUIPMENT

The lu numbers which are preceded by asterisks refer to devices which are preset to be the standard devices during the execution of the initializer until the PP message appears.

lun	Device	Equipment Number
*1	1777	1
2	1728-430 or 1729-2	10
3	1731-601	7(A/Q channel)
*4	1738-853/854	3
5	1751	3
*6	1711/1712/1713	1
7	1740-501	F
8	Dummy	
10	1726-405	4

2.3 SYSTEM UNIT

The standard system defines system units in LOG1A as follows:

System Unit	lun
Input comment	4
Output comment	4
Binary input	2
Binary output	3
List	9
Library	8
Scratch	8

FIELD CHANGE ORDER (FCO) LEVELS 3

The 1700 MSOS 3.0 operating system is checked on a computer system which has certain field change orders (FCO) installed. All FCO's issued by 12.1.70 were installed on this test system.

INITIALIZER CONTROL STATEMENTS 4

Through the use of these statements, it is possible to incorporate control statements with the actual binary programs.

4.1 *V ENTER STATEMENTS ON INPUT DEVICE

The *V statement instructs the system initializer to obtain subsequent control statements from the input device.

4.2 *U ENTER STATEMENTS ON COMMENT DEVICE

The *U statements instruct the system initializer to obtain its next and subsequent control statements from the comment device. This statement remains in effect until a *V statement is entered from the binary input device. The *U may be used to return control to the teletypewriter wherever options may be considered (loading a special routine from another device, deleting programs, etc.).

4.3 *S ASSIGN ENTRY POINT NAME

*S patches external strings at system initialization time. It permits the name n to be assigned a value and to be placed in the loader table as an entry point. The *S statements may be used to define unpatched externals to eliminate the error printout on the listing (e.g., *S, THREE, 7FFF). The *S can also cause a program to be eliminated by doubly defining an entry point (e.g., *S, PRINT1, 7FFF). This is useful in modifying a system when the source is magnetic tape or disk.

4.3.1 *S,n,hhhh

This statement assigns the hexadecimal value hhhh to the entry point name n and places both in the loader table. Previously defined external strings are patched with hhhh as are future references.

4.3.2 *S,n,S

This statement assigns the value of the current mass storage sector to the entry point name n. This statement permits dynamic assignment of values to symbolic names.

4.3.3 *S,n,P

This statement assigns the current value of the program base to the entry point name n. The program base is the next available core location into which the initializer loads.

INITIALIZER PROCEDURES 5

The following procedures describe how to enter data into core memory, examine data in core memory, and execute an instruction sequence. These procedures are incorporated into the system initialization instructions detailed in part I, section 3.

5.1 ENTERING DATA INTO CORE MEMORY

1. Press master CLEAR switch on console
2. Set all switches to the neutral positions
3. Set SELECTIVE STOP switch
4. Set P register
5. Set push button register to the core location into which the first word is to be stored
6. Set ENTER/SWEEP switch to ENTER
7. Set X register
8. Enter first (or next) word of code into push button register
9. Momentarily move RUN/STEP switch to STEP
10. Clear the X register by pressing the display CLEAR button
11. Repeat steps eight through ten for all words to be entered
12. Release SELECTIVE STOP switch when finished

5.2 EXAMINING DATA IN CORE MEMORY

1. Press master CLEAR switch on console
2. Set all switches to the neutral positions
3. Set SELECTIVE STOP switch
4. Set the P register
5. Set the push button register to the first core location to be examined
6. Set the X register
7. Set the ENTER/SWEEP switch to SWEEP
8. Momentarily move the RUN STEP switch to STEP
9. The data in the core location entered into the P register above will be displayed on the push button register
10. Repeat step eight to display the next sequential word of core memory
11. Release SELECTIVE STOP switch when finished

5.3 EXECUTING INSTRUCTION SEQUENCE

1. Press master CLEAR switch on console
2. Set all switches to neutral position
3. Set the P register
4. Enter the core location for the first instruction of the sequence into the push button register
5. Set the A, Q, and X registers to their specified contents
6. Set the SELECTIVE STOP and/or the SELECTIVE SKIP switches if necessary
7. Momentarily set the RUN STEP switch to RUN

INSTALLATION MESSAGES 6

6.1 SYSTEM INITIALIZER MESSAGES

SI	It informs the operator on the comment medium that the system initializer is ready to begin operation.
Q	Informs the operator (on comment medium) that system initializer is ready to accept another control statement.
L, nn FAILED ACTION	Appears when a driver cannot recover from an error. The operator can then take corrective action and respond with either RP or CU. RP causes the request to be repeated. CU causes the error condition to be reported to the program which made the request. Any other entry causes ACTION to be retyped.
ERROR 1	Asterisk initiator missing
ERROR 2	Number appears in name field
ERROR 3	Illegal control statement
ERROR 4	Input mode illegal
ERROR 5	No further *YM statements can be entered
ERROR 6	No further *Y statements can be entered
ERROR 7	*F statement previously entered
ERROR 8	Name appears in number field
ERROR 9	Illegal HEX core relocation field
ERROR A	Illegal mass storage sector number
ERROR B	Error return from loader module
ERROR C	Unpatched external at conclusion of *M load
ERROR D	Unpatched external at conclusion of *L load
ERROR E	Field terminator invalid
ERROR F	More than 120 characters in control statement
ERROR 10	Ordinal name without ordinal number
ERROR 11	Doubly defined entry point
ERROR 12	Invalid ordinal number
ERROR 13	*F statement not previously entered
ERROR 14	Data declared during *M load but not by first segment. Initialization restarted
ERROR 15	Attempt made to enter data into location 0 or above location $FE. Initialization restarted
ERROR 16	Irrecoverable mass storage I/O error
ERROR 17	Irrecoverable error. Last program loaded was ignored

6.2 PROGRAM LOADING MESSAGES

All loading error messages appear on the standard print output device.

E01	Irrecoverable input error; causes termination.
E03	Illegal or out-of-order input block; causes termination of load. This diagnostic also appears on the comment device when illegal input from that device is detected. The device is interrogated for a new statement.
E04	Incorrect common block storage reservation. Occurs if the largest common storage declaration is not on first NAM block to declare common storage. The loader uses the previously declared length and continues.
E05	Program too long or loader table overflow. Terminates loading. Occurs if program to be loaded exceeds available unprotected core. It may be possible to load the program by re-arranging the order of loading to insure entry points are defined before they are referred to as external symbols. Loader produces a memory map and list of unpatched externals prior to terminating the load.
E06	Attempt to load information in protected core; causes termination of load.
E07	Attempt to begin data storage beyond assigned block; causes termination of load.
E08	Duplicate entry point; loading is terminated.
E10	Unpatched external. External name is printed following diagnostic. When all unpatched externals have been printed, the operator may terminate the job or continue execution regardless of unpatched externals.
E11	Error in HEX block; loader skips remainder of block and resumes loading with the next block. The starting address is printed following diagnostic.
E12	Two programs reference same external name; one with absolute addressing, the other with relative addressing; loading is terminated.
E13	Undefined or missing transfer address; this code is not given if the loading operation is part of system initialization. Occurs when loader does not encounter a name for the transfer address or the name encountered is not defined in loader's table as entry point name.
E14	Loader request operation code word illegal.
E15	Address in I2 table is greater than $FE; issued only during system initialization. The post-resident loader initializer, I2 contains a table of information designated for locations within the communication region. An entry in this table consists of the storage address and the constant to be stored. If the address is greater than $FE, this comment is printed.

6.3 JOB PROCESSING MESSAGES

PARITY, hhhh — Memory parity error at location $hhhh_{16}$. Message appears on output comment device.

OV — Overflow of volatile storage. Message appears on output comment device.

ER — Unintelligible control statement following a manual interrupt command.

L, nn FAILED, code — Informs operator of device failure.

ACTION — Requests operator action when a failed device has no alternate. The device is identified in the FAILED diagnostic.

nn — logical unit number

code — code indicating cause of failure as follows. These are typical error codes. See the individual driver for the actual error codes.

00	I/O hangup	07	Echo check error on punch operation
01	Internal or external reject	08	Illegal Hollerith punch
02	Alarm	09	Sequence error
03	Parity error	10	Non-negative record length
04	Checksum error	11	Change from read mode to punch mode or vice versa
05	Internal reject	12	No $\frac{7}{9}$ punch
06	External reject	13	Error in disk read of mass memory driver

ALT, aa — Informs operator an alternate device, aa, has been assigned.

J01, hhhh — Program protect violation. hhhh is current contents of P register. Standard comments device.

J02, hhhh — Illegal request or parameters at location $hhhh_{16}$. Standard comment device.

J03, statement — Unintelligible control statement is output with the diagnostic. Standard comments device.

J04, statement — Illegal or unintelligible parameters in control statement. Standard comments device.

J05 — Statement entered after manual interrupt illegal. Standard comment device.

J06, hhhh — A threadable request was made at level one when no protect processor stack space was available, or an unprotected threaded request was made at level one. Standard comments device.

J07, hhhh — Unprotected program tried to access protected device from location hhhh. Standard comments device.

J08, hhhh — Attempt to access read only unit for write, or write only unit for read, or an attempt to access an unprotected request on a protected unit. Standard comments device.

6.4 DEBUGGING AND LIBRARY EDITING MESSAGES

The following messages appear on the output comment device. Both the system initializer and LIBEDT will attempt error recovery whenever possible. Illegal input statements are not processed.

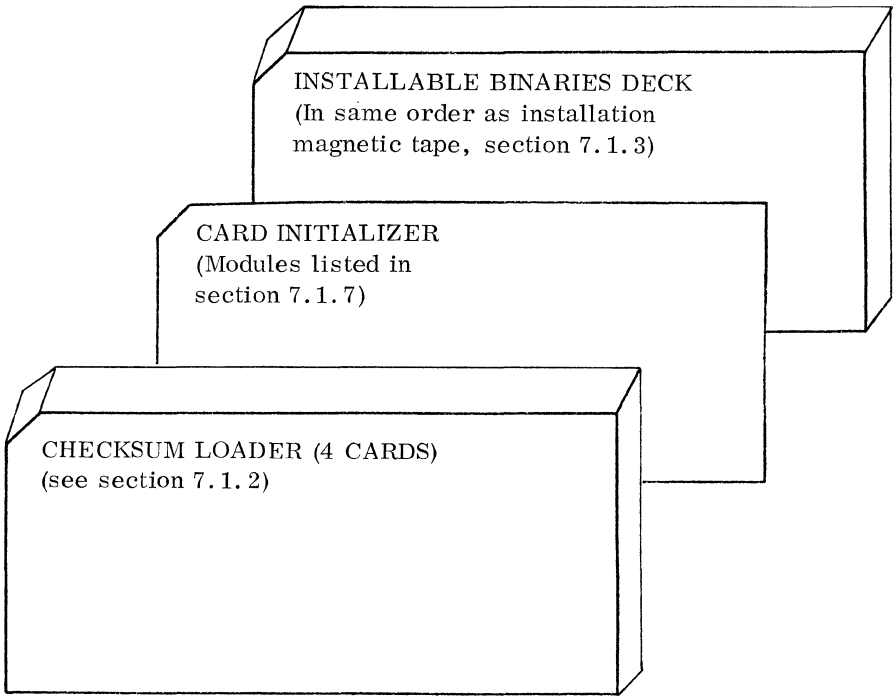
BP, hhhh	Breakpoint program ready for input. The breakpoint address $hhhh_{16}$ is printed only if breakpoint program was entered from previously set breakpoint.
B01, statement	Statement or parameters are unintelligible for the breakpoint program.
B02, hhhh	$hhhh_{16}$ cannot be processed by breakpoint program because it is protected.
B03, hhhh	Breakpoint limit exceeded. $hhhh_{16}$ is the last breakpoint processed.
B04	Previous *E statement requested entries in protected core. Entries are not processed; breakpoint program waits for new statement.
RE	Recovery program ready to accept control statements.
LIB	Library editing program ready to accept control statements.
J	Job processor waiting for control statement from input comment device.
L01	More than six characters in a parameter name presented to the library editing program.
L02	More than 6 digits in a number presented to the library editing program.
L03	Improper system directory ordinal presented to the library editing program.
L04	Invalid control statement presented to the library editing program.
L05	Illegal field delimiter in a control statement presented to the library editing program.
L06	Illegal field in control statement presented to the library editing program.
L07	Errors in loading as a result of a library editing program control statement.
L08	A program to be added to the program library has an entry point duplicating one already in the directory.
L09	Standard input failed on first input record following an *N request.
L10	The operator is deleting a program which is not in the library.
L11	No header record on file input from mass storage.
L12	On an *L, entry statement, either there was an input error, or the first record was not a NAM block.
L13	Common declared by the program being loaded exceeds available common.
L14	Program being loaded is longer than the size of unprotected core, but not longer than the distance from the start of unprotected core to the top of core.
L15	Illegal input block encountered; last program stored in library is not complete.
L16	I/O input error occurred; last program stored is not complete.
L17	*L program being installed exceeds the capacity of LIBEDT to input from mass storage.
L18	Attempt to load a zero length program during an *M request.

CARD DECK AND TAPE CONTENTS 7

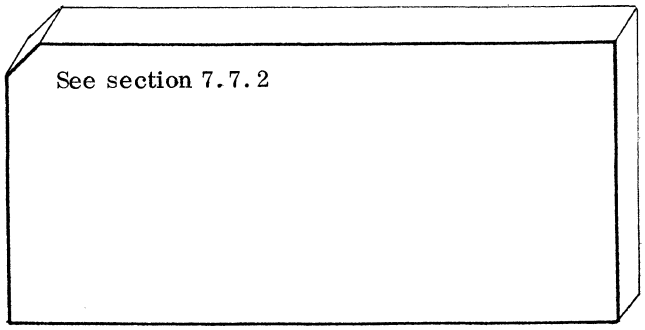
7.1 MSOS 3.0

7.1.1 STRUCTURES

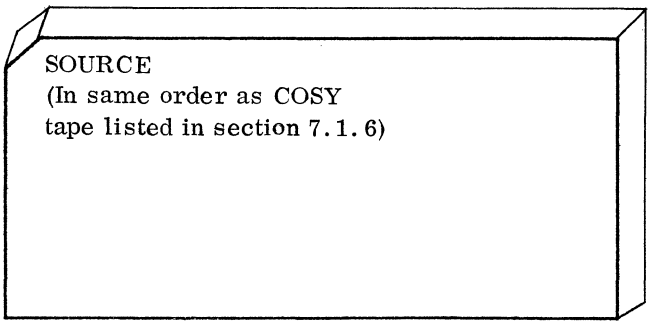
Card Installation Deck



System Definitions and Skeletons Card Deck



Optional Card Source Deck



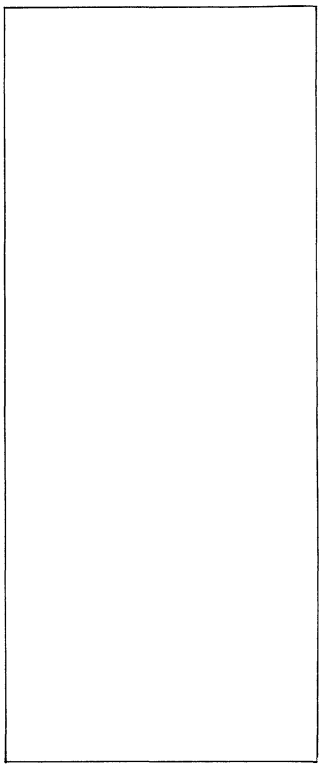
Magnetic Installation Tape

See section 7.1.3 for a list of the tape contents; section 7.1.7 lists and identifies MSOS 3.0 modules.

ABSOLUTIZED INITIALIZER
SYSBUF
ENGINEERING FILE
LIBEDT
STANDARD SYSTEM
LOADER
JOB PROCESSOR
ODEBUG
BRKPT
DRIVERS
VRFCTN
REQUEST PRIORITY ASSIGNMENTS
SMR
EOF

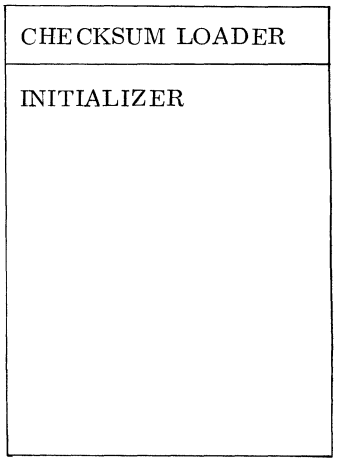
System Definitions and Skeletons Magnetic Tape

See section 7.7.2.



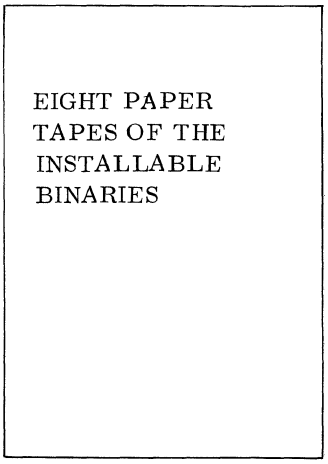
Paper Tape Initializer

See section 7.1.4 for a listing of the initializer tape.



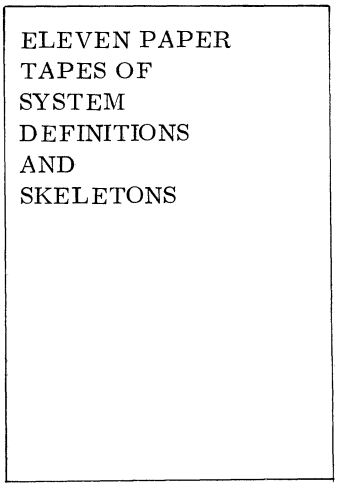
Paper Installation Tapes

See section 7.1.5 for a list of tape contents.



System Definitions and Skeletons Paper Tape

See section 7.7.2.



Optional Tapes

COSY SOURCE MAGNETIC TAPE	LIST 1 MAGNETIC TAPE	LIST 2 MAGNETIC TAPE	LIST 3 MAGNETIC TAPE
See section 7.1.6	The three list tapes are in the same order as the COSY tape (section 7.1.6) with the following divisions.		
CORE RESIDENT	CONTAINS CORE RESIDENT THROUGH MAS DMP	CONTAINS BRKPT THROUGH DRIVERS	CONTAINS RE-ENTRANT FORTRAN
JOB PROCESSOR			ENCODE DECODE
ODEBUG			SMR
LOADER			
MASS MEMORY MODULES			
BRKPT			
DRIVERS			
RE-ENTRANT AND NON-RE-ENTRANT FORTRAN AND ENCODE/DECODE			
SMR			
EOF			

7.1.2 CARD INSTALLATION DECK

The card installation deck contains three parts. Part I is the checksum loader:

```
                         NAM   LOADCR
                         ENT   LOADCR
P0000 6867   LOADCR STA*  STADD
P0001 0C06           ENQ   6
P0002 0844           CLR   A
P0003 6A67   OUT     STA*  CARD,Q
P0004 0142           SQZ   ZERO
P0005 0DFE           INQ   -1
P0006 18FC           JMP*  OUT
P0007 0804   ZERO    SET   A
P0008 685E           STA*  COUNT
P0009 0844   READCR  CLR   A
P000A 685F           STA*  QCNT
P000B 685F           STA*  CARD
P000C E000           LDQ   =N$0421
P000D 0421
P000E C000           LDA   =N$0080
P000F 0080
P0010 03FE           OUT   -1
P0011 E000   LOAD    LDQ   =N$0420
P0012 0420
P0013 02FF           INP   -1
P0014 E855           LDQ*  QCNT
P0015 6A68           STA*  READBF,Q
P0016 D853           RAO*  QCNT
P0017 D853           RAO*  CARD
P0018 C851           LDA*  QCNT
P0019 09FB           INA   -4
P001A 0101           SAZ   CLEAR
P001B 18F5           JMP*  LOAD
P001C 0842   CLEAR   CLR   Q
P001D 484C           STQ*  QCNT
P001E CA5F   PAK     LDA*  READBF,Q
P001F 0FC4           ALS   4
P0020 684D           STA*  TEMP
P0021 0D01           INQ   1
P0022 CA5B           LDA*  READBF,Q
P0023 A000           AND   =N$0F00
P0024 0F00
P0025 0F48           ARS   8
P0026 584B           RTJ*  FIGR
P0027 D842           RAO*  QCNT
P0028 CA55           LDA*  READBF,Q
P0029 A000           AND   =N$00FF
P002A 00FF
P002B 0FC8           ALS   8
P002C 6841           STA*  TEMP
P002D 0D01           INQ   1
P002E CA4F           LDA*  READBF,Q
P002F A000           AND   =N$0FF0
P0030 0FF0
P0031 0F44           ARS   4
P0032 583F           RTJ*  FIGR
```

```
P0033 D836           RAO* QCNT
P0034 CA49           LDA* READBF,Q
P0035 A000           AND  =N$000F
P0036 000F
P0037 0FCC           ALS  12
P0038 0D01           INQ  1
P0039 6834           STA* TEMP
P003A CA43           LDA* READBF,Q
P003B 0D01           INQ  1
P003C A000           AND  =N$0FFF
P003D 0FFF
P003E 5833           RTJ* FIGR
P003F C827   XFER    LDA* COUNT
P0040 0C01           ENQ  1
P0041 0124           SAP  SKIP
P0042 E83C           LDQ* READBF+1
P0043 482B           STQ* RECNT
P0044 D822           RAO* COUNT
P0045 0C02           ENQ  2
P0046 C824   SKIP    LDA* CARD
P0047 09FB           INA  -4
P0048 0102           SAZ  STORE
P0049 0842           CLR  Q
P004A 1808           JMP* DATA
P004B C832   STORE   LDA* READBF
P004C 981F           SUB* SEQNCE
P004D 0132           SAM  TWO
P004E 681D           STA* SEQNCE
P004F 1803           JMP* DATA
P0050 0804   TWO     SET  A
P0051 681B           STA* ERROR
P0052 CA2B   DATA    LDA* READBF,Q
P0053 6C14           STA* (STADD)
P0054 C81C           LDA* WDCNT
P0055 8819           ADD* RECNT
P0056 D81A           RAO* WDCNT
P0057 0111           SAN  1
P0058 1822           JMP* CHECK
P0059 D80E           RAO* STADD
P005A 0814           TRQ  A
P005B 09FD           INA  -2
P005C 0122           SAP  LOOP
P005D 0D01           INQ  1
P005E 18F3           JMP* DATA
P005F C80B   LOOP    LDA* CARD
P0060 09AF           INA  -80
P0061 0123           SAP  3
P0062 0844           CLR  A
P0063 6806           STA* QCNT
P0064 18AC           JMP* LOAD
P0065 18A3           JMP* READCR
P0066 FFFF   COUNT   NUM  -0
P0067 0000   STADD   NUM  0
```

```
P0068 0000    QSAV    NUM   0
P0069 0000    QCNT    NUM   0
P006A 0000    CARD    NUM   0
P006B 0000    SEQNCE  NUM   0
P006C 0000    ERROR   NUM   0
P006D 0000    TEMP    NUM   0
P006E 0000    RECNT   NUM   0
P006F 0000    CKSUM   NUM   0
P0070 0000    WDCNT   NUM   0
P0071 0B00    FIGR    NOP   0
P0072 48F5            STQ*  QSAV
P0073 E8F5            LDQ*  QCNT
P0074 B8F8            EOR*  TEMP
P0075 6A08            STA*  READBF,Q
P0076 88F8            ADD*  CKSUM
P0077 68F7            STA*  CKSUM
P0078 E8EF            LDQ*  QSAV
P0079 1CF7            JMP*  (FIGR)
P007A E8F4    CHECK   LDQ*  CKSUM
P007B C8F0            LDA*  ERROR
P007C 0000            SLS   0
P007D 0004            BZS   READBF(4)
                      END   LOADCR
```

Part 2 contains the absolutized initializer. The initializer modules are listed in section 7.1.7.

Part 3 contains the relocatable installable binaries of the system in the same order as they appear on the magnetic installation tape (section 7.1.3).

7.1.3 MAGNETIC INSTALLATION TAPE

Record one contains the initializer modules listed in section 7.1.7.

```
*S,SYSLVL,5245
*S,DTIMER,7FFF
*S,SYSCOP,7FFF
*S,ONE,7FFF
*S,TWO,7FFF
*S,THREE,7FFF
*S,WDADR,1
*YM,EFILE,1
*YM,LIBEDT,2
*YM,LOADSD,3
*YM,JOBENT,4
*YM,JOBPRO,5
*YM,PROTEC,6
*YM,JPLOAD,7
*YM,JPCHGE,8
*YM,JPT13,9
```

```
*YM,MIPRO,10
*YM,RESTOR,11
*YM,ODEBUG,12
*YM,RCOVER,13
*YM,BRKPT,14
*YM,SELF,15
*YM,LOGGER,16
*L     LOCORE
         LOCORE
         SYSBUF
         TRVEC
         DRCORE
         MIPROC
*M         EFTLE
         EF
*M         LI-EDT
         LIBEDT
*L     SCHEDU
         SCHEDU
         MDISP
         NCMPRO
         NENR
         ADEV
*M         LOADSD
         LOAD
         BRANCH
         LIDRIV
         LCDRIV
         LMDRIV
         LLDRIV
         SCAN
         CHPU
         ADJOVF
         CONVRT
         TABSCH
         TABSTR
         LSTOUT
         LINK1
         LINK2
         COREXT
         DPRADD
         LOADER
         NAMPRO
         PBDB/S
         ENTEXT
         XFRPRO
         HEXPRO
         EOLPRO
         ADRPRO
```

```
*L      ALCORE
          ALCORE
          ALVOL
          OFVOL
          PARAME
          COMMON
          NEPROC
          NMONI
          RW
          MAKO
          MINT
*M          JOBENT
          JOBENT
          T11
          T7
          T5
          T3
*M          JOBPRO
          JOBPRO
*M          PROTEC
          PROTEC
          JBKILL
*M          JPLOAD
          JPLOAD
*M          JPCHGE
          JPCHGE
          ASCHEX
*M          JPT13
          T13
*M          MIPRO
          MIPRO
*M          RESTOR
          RESTOR
*M          ODEBUG
          ODEBUG
*M          RCOVER
          RCOVER
          OUTSEL
          DMPCOR
          MASDMP
*M          BRKPT
          BRKPTD
          BIASCI
          SIFT
          RETJMP
          JUMPTO
          ENTER
          ENTCOR
          PRTREG
          SETBRP
          TERMIN
          DMPCOR
          MASDMP
          RESUME
*M          SELF
```

```
         SELFS
*M         LOGGER
         LOGA
*L     MASDRV
         MASDRV
         S13001
*M     MASS MEMORY DRIVERS
         S13002
*S,M1713R,S
*M
         S13003
*S,M1713P,S
*M
         PRT40
*S,MAS501,S
*M
         DR1728
*S,MAS28,S
*M
         CD1729
*S,MAS292,S
*M
         CR405
*S,MAS405,S
*M
         DR1732
*S,MAS32,S
*M
         PUNCDR
*S,TP1777,S
*M
         PTREAD
*S,TR1777,S
*L
         TAPEDR
         FRWA
         FRWB
         RWBA
         RECOVT
         TAPE
         DISKWD
         SPACE
*S,MAS31,7FFF
*S,TIVALU,7FFF
*S,ABUFAL ,7FFF
*S,SNAPE,7FFF
*S,PARITY,7FFF
*S,IPROC1,7FFF
*S,T30,7FFF
*S,T29,7FFF
*S,T28,7FFF
*S,T27,7FFF
*S,T26,7FFF
*S,T25,7FFF
*S,T24,7FFF
*S,T23,7FFF
*S,T22,7FFF
*S,T21,7FFF
*S,T20,7FFF
*S,T19,7FFF
```

```
*S,T18,7FFF
*S,T17,7FFF
*S,T16,7FFF
*S,T13,7FFF
*S,T11,7FFF
*S,T8,7FFF
*S,T7,7FFF
*S,T5,7FFF
*S,T3,7FFF
*S,DEBUG,7FFF
*S,TIMACK,7FFF
*F
        VRFCTN
*T
*S,1,0,M
*S,2,3,M
*S,3,0,M
*S,4,1,M
*S,5,2,M
*S,6,3,M
*S,7,2,M
*S,8,2,M
*S,9,2,M
*S,10,2,M
*S,11,2,M
*S,12,3,M
*S,13,3,M
*S,14,3,M
*S,15,3,M
*S,16,3,M
*U
*K,I6,P8
*P,F
        DSKTAP
        EQCODE
        MTINP
        MTOUT
        DCODHX
        CDRIVE
        MDRIVE
*T
*K,I8
*N,DSKTAP,,,B
*K,I6
*L,DTLP
        DTLP
*L,UPDATE
        UPDATE
*L,LISTR
        LISTR
*L,CYFT
        CYFT
*L,LCOSY
        LCOSY
*L,SILP
        SILP
*L,OPSORT
        OPSORT
*K,I6,P8
*P,F
```

```
          CONTRL
          LIB
          IDRIV
          MTIDRV
          PTIDRV
          CDIDRV
          MDRIV
          MSDISK
          MSDRUM
          I2
          I2DISK
          I2DRUM
          I1
          ILOAD
          CDRIV
          LPRINT
*T
*K,I8
*N,SI,,,8
*U
```

7.1.4 PAPER TAPE INITIALIZER

The initializer MSOS 3.0 release tape contains two format records: record 1 is the checksum loader; record 2 is the system initializer.

Record 1 Checksum Loader

```
P0000  6815          STA*   LOC
P0001  0A20          ENA    $20
P0002  E000          LDQ    =N$A1
P0003  00A1
P0004  03FE          OUT    -1
P0005  5823          RTJ*   WORD1
P0006  6811   BACK     STA*   IT
P0007  5811   LOOP     RTJ*   WORD
P0008  6C0D          STA*   (LOC)
P0009  D80C          RAO*   LOC
P000A  D80D          RAO*   IT
P000B  C80C          LDA*   IT
P000C  0101          SAZ    1
P000D  18F9          JMP*   LOOP
P000E  580A          RTJ*   WORD
P000F  0000          SLS    0
P0010  4807          STQ*   IT
P0011  0181          SWS    1
P0012  1C05          JMP*   (IT)
P0013  5C04          RTJ*   (IT)
P0014  0000          SLS    0
P0015  0001          BZS    LOC, CHKSUM, IT
P0016  0001
P0017  0001
P0018  0B00   WORD     NOP    0
P0019  5809          RTJ*   GET
P001A  0FC8   WORD2    ALS    8
P001B  680D          STA*   TEMP
P001C  5806          RTJ*   GET
P001D  B80B          EOR*   TEMP
P001E  0822          TRA    Q
P001F  F8F6          ADQ*   CHKSUM
P0020  48F5          STQ*   CHKSUM
P0021  1CF6          JMP*   (WORD)
P0022  0B00   GET      NOP    0
P0023  E000          LDQ    =N$A0
P0024  00A0
P0025  0A00          ENA    0
P0026  02FE          INP    -1
P0027  1CFA          JMP*   (GET)
P0028  0B00   WORD1    NOP    0
P0029  0844          CLR    A
P002A  68EB          STA*   CHKSUM
```

```
P002B   C8FC            LDA*    WORD1
P002C   68EB            STA*    WORD
P002D   58F4    OVER    RTJ*    GET
P002E   0111            SAN     1
P002F   18FD            JMP*    OVER
P0030   18E9            JMP*    WORD2
        0028  P TEMP    EQU     TEMP (WORD1)
                        END
```

Record 2 System Initializer

Record 2 contains the system initializer modules absolutized in absolute binary records by using the *P function of the LIBEDT routine. The system initializer modules are listed in section 7.1.7.

7.1.5 INSTALLATION PAPER TAPES

Installation Paper Tape 1

```
*S,SYSLVL,5245
*S,DTIMER,7FFF
*S,SYSCOP,7FFF
*S,ONE,7FFF
*S,TWO,7FFF
*S,THREE,7FFF
*S,WDADR,1
*YM,EFILE,1
*YM,LIBEDT,2
*YM,LOADSD,3
*YM,JOBENT,4
*YM,JOBPRO,5
*YM,PROTEC,6
*YM,JPLOAD,7
*YM,JPCHGE,8
*YM,JPT13,9
*YM,MIPRO,10
*YM,RESTOR,11
*YM,ODEBUG,12
*YM,RCOVER,13
*YM,BRKPT,14
*YM,SELF,15
*YM,LOGGER,16
*L     LOCORE
         LOCORE
         SYSBUF
         TRVEC
         DRCORE
         MIPROC
*M         EFILE
         FF
```

Installation Paper Tape 2

```
*M         LIBEDT
         LIBEDT
*L     SCHEDU
         SCHEDU
         NDISP
         NCMPRQ
         NFNR
         ADEV
```

Installation Paper Tape 3

```
*M      LOADSD
        LOAD
        BRANCH
        LIDRIV
        LCDRIV
        LMDRIV
        LLDRIV
        SCAN
        CHPU
        ADJOVF
        CONVRT
        TABSCH
        TABSTR
        LSTOUT
        LINK1
        LINK2
        COREXT
        DPRADD
        LOADER
        NAMPRO
        RBDB7S

        ENTEXT
        XFRPRO
        HEXPRO
        FOLPRO
        ADRPRO
```

Installation Paper Tape 4

```
*L      ALCORE
          ALCORE
          ALVOL
          OFVOL
          PARAME
          COMMON
          NEPROC
          NMONT
          RW
          MAKQ
          MINT
*M          JORENT
          JORENT
          T11
          T7
          T5
          T3
*M          JORPRO
          JORPRO
*M          PROTEC
          PROTEC
          JBKILL
*M          JPLOAD
          JPLOAD
*M          JPCHGE
          JPCHGE
          ASCHEX
*M          JPT13
          T13
*M          MIPRO
          MIPRO
```

Installation Paper Tape 5

```
*M          RESTOR
           RESTOR
*M          ODEBUG
           ODEBUG
*M          RCOVER
           RCOVER
           OUTSEL
           DMPCOR
           MASDMP
*M          BRKPT
           BRKPTD
           BIASCI
           SIFT
           RETJMP
           JUMPTO
           ENTER
           ENTCOR
           PRTREG
           SETBRP
           TERMIN
           DMPCOR
           MASDMP
           RESUME
*M          SELF


           SELFS
*M          LOGGER
           LOGA
```

Installation Paper Tape 6

```
*L     MASDRV
         MASDRV
         S13001
*M     MASS MEMORY DRIVERS
         S13002
*S,M1713R,S
*M
         S13003
*S,M1713P,S
*M
         PRT40
*S,MAS501,S
*M
         DR1728
*S,MAS28,S
*M
         CD1729
*S,MAS292,S
*M
         CR405
*S,MAS405,S
*M
         DR1732
*S,MAS32,S
*M
         PUNCDR
*S,TP1777,S
*M
         PTREAD
*S,TR1777,S
```

```
*L
         TAPEDR
         FRWA
         FRWB
         RWBA
         RECOVT
         TAPE
         DISKWD
         SPACE
*S,MAS31,7FFF
*S,TIVALU,7FFF
*S,ABUFAL,7FFF
*S,SNAPE,7FFF
*S,PARITY,7FFF
*S,IPROC1,7FFF
*S,T30,7FFF
*S,T29,7FFF
*S,T28,7FFF
*S,T27,7FFF
*S,T26,7FFF
*S,T25,7FFF
*S,T24,7FFF
*S,T23,7FFF
*S,T22,7FFF
*S,T21,7FFF
*S,T20,7FFF
*S,T19,7FFF
*S,T18,7FFF
*S,T17,7FFF
*S,T16,7FFF
*S,T13,7FFF
*S,T11,7FFF
*S,T8,7FFF
*S,T7,7FFF
*S,T5,7FFF
*S,T3,7FFF
*S,DEBUG,7FFF
*S,TIMACK,7FFF
*F
         VRFCTN
*T
*S,1,0,M
*S,2,3,M
*S,3,0,M
*S,4,1,M
*S,5,2,M
*S,6,3,M
*S,7,2,M
*S,8,2,M
*S,9,2,M
*S,10,2,M
*S,11,2,M
*S,12,3,M
*S,13,3,M
*S,14,3,M
*S,15,3,M
*S,16,3,M
```

(continued in next column)

```
*U
*K,I6,P8
*P,F
         DSKTAP
         EQCODE
         MTINP
         MTOUT
         DCODHX
         CDRIVE
         MDRIVE
*T
*K,I8
*N,DSKTAP,,,B
```

```
*K,I6
*L,DTLP
        DTLP
*L,UPDATE
        UPDATE
*L,LISTR
        LISTR
*L,CYFT
        CYFT
*L,LCOSY
        LCOSY
*L,SILP
        SILP
*L,OPSORT
        OPSORT
*K,I6,P8
*P,F
```

```
        CONTRL
        LIB
        IDRIV
        MTIDRV
        PTIDRV
        CDIDRV
        MDRIV
        MSDISK
        MSDRUM
        I2
        I2DISK
        I2DRUM
        I1
        ILOAD
        CDRIV
        LPRINT
*T
*K,I8
*N,SI,,,B
*U
```

7.1.6 OPTIONAL COSY SOURCE TAPE

LOCORE
SYSBUF
SCHEDU
NDISP
NCMPRQ
NFNR
ADEV
BUFFER
RDISP

EFILE
LOGA
SELF
CONTRL
LIB
IDRIV
MTIDRV
PTIDRV
CDIDRV
MDRIV
MSDISK
MSDRUM
I2
I2DISK
I2DRUM
I1
ILOAD
CDRIV
LPRINT

DRCORE
ALCORE
ALVOL
OFVOL
TRVEC
PARAME
COMMON
NIPROC
NEPROC
NMONI
RW
MAKQ
MINT
TMINT
DTMER
SPACE

JOBENT
T11
T7
T5
T3
JOBPRO
PROTEC
JBKILL
JPLOAD
JPCHGE
ASCHEX
T13
MIPRO
RESTOR
ODEBUG

LOAD
BRANCH
LIDRIV
LCDRIV
LMDRIV
LLDRIV
SCAN
CHPU
ADJOVF
CONVRT
TABSCH
TABSTR
LSTOUT
LINK1
LINK2
COREXT
DPRADD
LOADER
NAMPRO
RBDBZS
ENTEXT
XFRPRO
HEXPRO
EOLPRO
ADRPRO

RCOVER
OUTSELF
DMPCOR
MASDMP

(continued in next column)

BRKPTD
SIFT
BIASCI
RETJMP
JUMPTO
ENTER
ENTCOR
PRTREG
TERMIN
RESUME

DPCORB
MSDMPB
SETBRP

LIBEDT

TELTYP
DRMDRZ
TAPDRB
RWBAB
FRWAB
FRWBB
RECVTB
TABEDR
FRWA
FRWB
RWBA
RECOVT
TAPE
DR1728
CD1729
DISKWD
DISK
BUFALC
MASDRV
DBLDRV
DR1732
PRT40
CR405
S13001
S13002
S13003
PUNCDR
PTREAD
STCK

Re-entrant FORTRAN

COSY Deck Name	Program Names
IOCODR	IOCODE
INITLR	INITAL
RSTORR	RSTORE
GETCHR	IGETCH
IPACKR	IPACK
UPDATR	UPDATE
DECPLR	DECPL
INTGRR	INTGR
SPACER	SPACEX
HOLR	HOLRTH
DCHXR	DCHX
HXASCR	HXASC
AFMTOR	AFRMOT
RFMTOR	RFRMOT
AFMTIR	AFRMIN
RFMTIR	RFRMIN
ASCHXR	ASCHX
HXDCR	HXDC
FLOTIR	FLOTIN
FOUTR	FOUT
EOUTR	EOUT
EWRITR	EWRITE
AFORMR	AFORM
RFORMR	RFORM
HEXAR	HEXASC
HEXDR	HEXDEC
ASCIIR	ASCII
DECHXR	DECHEX
FLOTGR	FLOATG
INTL1R	INITL1
FORMTR	FORMTR
Q8QFIR	Q8QFI
Q8QFLR	Q8QFL
Q8QFXR	Q8QFX
Q8QIOR	Q8QIO
Q8QGTR	Q8QGTX

Non-re-entrant

COSY Deck Name	Program Names
IOCODE	IOCODE
IGETCH	IGETCH
IPACK	IPACK
UPDATN	UPDATE
DECPL	DECPL
INTGR	INTGR
SPACEN	SPACEX
HOLRTH	HOLRTH
DCHX	DCHX
HXASC	HXASC
AFRMOT	AFRMOT
RFRMOT	RFRMOT
AFRMIN	AFRMIN
RFRMIN	RFRMIN
ASCII	ASCII
HXDC	HXDC
FLOTIN	FLOTIN
FOUT	FOUT
EOUT	EOUT
EWRITE	EWRITE
AFORM	AFORM
RFORM	RFORM
HEXASC	HEXASC
HEXDEC	HEXDEC
DECHEX	DECHEX
FLOATG	FLOATG
FORMTR	FORMTR
INITL1	INITL1
ASCHX	ASCHX
PSEUDO	PSEUDO
Q8QFI	Q8QFI
Q8QFL	Q8QFL
Q8QFX	Q8QFX
Q8QIO	Q8QIO
Q8QGTX	Q8QGTX

Re-entrant

COSY Deck Name	Program Names
FORTR	FORTRA

Non-re-entrant

FORTN	FORTRA
Q8EXPN	Q8EXPN
Q8PRMN	Q8PRMS
Q8AB	Q8AB
IFALT	IFALT
SIGN	SIGN
FXFL	FXFL
EXPPRG	EXPPRG
SQRTF	SQRTF
LNUPRG	LNUPRG
TANH	TANH
SINCOS	SINCOS
ARCTPG	ARCTPG
FLOAT	FLOAT

Re-entrant

Q8EXPR	Q8EXPR
Q8PRMR	Q8PRMS
Q8ABR	Q8ABR
IFALTR	IFALTR
SIGNR	SIGNR
FXFLR	FXFLR
EXPRGR	EXPRGR
SQRTFR	SQRTFR
LNPRGR	LNPRGR
TANHR	TANHR
SNCSR	SNCSR
ARCPGR	ARCPGR
FLOATR	FLOATR

SMR

COSY Deck Name	Program Names
OPSORT	OPSORT
UPDAT	UPDATE
CYFT	CYFT
LISTR	LISTR
LCOSY	LCOSY
SILP	SILP
DTLP	DTLP
DSKTAP	DSKTAP
DSKEQC	EQCODE
DSKMTI	MTINP
DSKMTO	MTOUT
DSKDHX	DCODHX
DSKCDR	CDRIVE
DSKMMD	MDRIVE

7.1.7 MSOS 3.0 MODULE LIST

System Initializer Modules

CONTROL†	Control module
LIB†	Library generation module
IDRIV†	Input control module (input device driver controller)
MDRIV†	Mass storage driver control module
CDRIV†	Comment control module (comment device driver)
ILOAD†	Resident loader (relocatable binary loading module)
I1†	Pre-resident load initialization
I2†	Initialization controller module 2; post-resident load initialization
MSDISK††	Pre-resident initialization 853/854 disk driver
MSDRUM††	Pre-resident initialization 1751 drum driver
I2 DISK††	Post-resident initialization disk driver
I2 DRUM††	Post-resident initialization drum driver
MTIDRV†††	601 magnetic tape driver
PTIDRV†††	1721/1722 paper tape reader driver
LPRINT†††	1742 line printer driver
CDIDRV	1728-430/1729-2/1726 card reader driver

Core Resident Modules

All of the following core resident modules are not included in a single system installation since this list includes all available core resident modules. The modules included in a particular system depend on the options desired.

† Required modules.
†† Normally either the disk or drum drivers are used, not both.
††† Optional according to the system configuration.

LOCORE †	16K LOCORE - predefined constants and interrupt slots
SYSBUF †	16K SYSBUF - system table
BUFFER	Software buffering package
DRCORE	Core allocator driver
ALCORE	Core allocator
SCHEDU	Scheduler
NDISP	Normal dispatcher
RDISP	Dispatcher for use of the re-entrant FORTRAN
ALVOL	Volatile storage allocator
OFVOL	Volatile storage overflow error reporting
TRVEC	Transfer vectors
PARAME	Parameter conversion routines
COMMON	Common interrupt handler
NIPROC	Normal internal interrupt handler
NEPROC	Normal external interrupt handler
NMONI	Normal monitor request processor
RW	Read/write request processor
NCMPRQ	Normal complete request module
NFNR	Normal find next request module
MAKQ	Read/write Q generation module
ADEV	Alternate device handler
MINT	Manual interrupt handler
TIMINT	Timer driver
DTIMER	Diagnostic timer module
SPACE†	Core structuring module

† These modules will change with customization.

Loader Modules

LOAD	Initialization
BRANCH	Call tape
LIDRIV	Input driver
LCDRIV	Comment driver
LMDRIV	Mass storage driver
LLDRIV	List driver
SCAN	Unpack input
CHPU	Unpack input
ADJOVF	15-bit arithmetic
CONVRT	Binary to ASCII conversion
TABSCH	EXT, ENT, search
TABSTR	Loader table generation
LSTOUT	Message output
LINK1	Linkage operation 1
LINK2	Linkage operation 2
COREXT	ABS output to mass storage
DPRADD	Single precision to double precision ADD
LOADER	Loader control
NAMPRO	NAM processor
RBDBZS	RBD, BZS processor
EXTENT	ENT, EXT processor
XFRPRO	XFR processor
HEXPRO	HEX processor
EOLPRO	EOL processor
ADRPRO	Address computation

Job Processor Modules

Mass Memory Module JOBENT:

JOBENT	JOB processor entry module
T11	Core request processor

T7	Loader request processor
T5	Exit request processor
T3	Status request processor

Mass Memory Module JOBPRO:

JOBPRO	Job processor control module

Mass Memory Module PROTEC:

PROTEC	Protect processor
JBKILL	Job kill module

Mass Memory Module JPLOAD:

JPLOAD	Loader

Mass Memory Module JPCHGE:

JPCHGE	Logical unit change module
ASCHEX	ASCII conversion module

Mass Memory Module JPT13:

T13	GTFILE request processor

Miscellaneous Mass Memory Modules

Mass Memory Module LIBEDT:

LIBEDT	Control module

Mass Memory Module MIPRO:

MIPRO	Manual interrupt processor

Mass Memory Module RESTOR:

RESTOR	Restores logical units

Mass Memory Module ODEBUG:

ODEBUG	On-line debug module

Mass Memory Module RCOVER:

RCOVER	Control module
OUTSEL	Output unit select module
DMPCOR	Core dump module
MASDMP	Mass storage dump module

Mass Memory Module BRKPT:

BRKPTD	Control module
SIFT	Statement analyzer
BIASCI	Binary ASCII conversion
RETJMP	*R statement processor
JUMPTO	*J statement processor
ENTER	*A, *Q, *J statement processor
ENTCOR	*E statement processor
PRTREG	*P statement processor
TERMIN	*T statement processor
RESUME	*C statement processor
DMPCOR	Core dump module
MASDMP	Mass memory dump module
SETBRP	*S statement processor

Driver Modules

TELTYP	1711/1712/1713 teletypewriter
DRMDRZ	1751 drum
TAPDRB	1731-1706-601 buffered magnetic tape control
RWBAB	1731-1706-601 buffered non-format read/write
FRWAB	1731-1706-601 buffered format ASCII read/write
FRWBB	1731-1706-601 buffered format binary read/write
RECVTB	1731-1706 recovery
TAPEDR	1731-601 magnetic tape control
FRWA	1731-601 format ASCII read/write
FRWB	1731-601 format binary read/write

RWBA	1731-601 non-format read/write
RECOVT	1731 recovery
TAPE	1731/1732 tape motion control
DR1728	1728-430 card reader driver
CD1729	1729-2 card reader driver
DISKWD	1738-853/854 disk word driver
DISK	1738-853/854 disk driver
MASDRV	Mass memory control program for drivers including 1713 teletypewriter
S13001	1713 teletypewriter keyboard module
S13002	1713 teletypewriter reader module
S13003	1713 teletypewriter punch module
DR1732	1732-608/609 magnetic tape
PRT40	1740/501 line printer driver
CR405	1726/405 card reader driver
PTREAD	1777 reader module
PUNCDR	1777 punch module
STCK	1777 status module
MASDRV	Mass memory driver module
DBLDRV	Mass memory driver module for two drivers
EFILE	Engineering file
LOGA	Engineering file
SELF	Engineering file
OPSORT	Operand sort
UPDATE	Binary update program
CYFT	COSY format
LISTR	List binary decks
LCOSY	List COSY
SILP	System initializer loader
DTLP	Disk to tape -- bootstrap
DSKTAP	Disk to tape -- control
EQCODE	Disk to tape -- equipment code processor

MTINP	Disk to tape -- magnetic tape input
MTOUT	Disk to tape -- magnetic tape output
DCODHX	Disk to tape -- decimal to hex conversion
CDRIVE	Disk to tape -- comment driver
MDRIVE	Disk to tape -- mass storage driver

7.2 MACRO ASSEMBLER 2.0

Installation Tapes

See section 7.2.1 for contents.

Optional Tapes

See section 7.2.2 for contents.

TAPE 1
PAPER TAPE

INSTALLATION
FROM PAPER
TAPE IN
RELOCATABLE
BINARY

INCLUDES

CONTROL
STATEMENTS
AND
ASSEM
PASS1
PASS2
PASS3
PASS4
AND
ABSOLUTIZED
MACSKL
AND
MACROS

OR

TAPE 4
MAGNETIC TAPE

INSTALLATION
FROM
MAGNETIC
TAPE IN
RELOCATABLE
BINARY

INCLUDES

CONTROL
STATEMENTS
AND
ASSEM
PASS1
PASS2
PASS3
PASS4
AND
ABSOLUTIZED
MACSKL
AND
MACROS

EOF

TAPE 2
PAPER TAPE

LIBRARY
MACRO
PREPARATION
PROGRAM
RELOCATABLE
BINARY OF
LIBMAC
LIBMC2
LIBMC3

TAPE 3
PAPER TAPE

SYSTEM
LIBRARY
MACROS
IN ASCII
SOURCE

Optional Tapes

SOURCE TAPE 5	LIST TAPE 6
COSY SOURCE OF:	LIST
ASSEM	
PASS1	
PA1 PR2	
PASS2	
PA2 PR2	
PASS3	
PA3 PR2	
PA3 PR3	
PASS4	
LIBMAC	
LIBMC2	
LIBMC3	
EOF	EOF

7.2.1 INSTALLATION TAPES

Tape 1 and Tape 4

Paper tape 1 and magnetic tape 4 are the same except for control statements which assign logical units.

```
*K,I6,P8 (I2 for the paper tape)
*L,ASSEM
    Relocatable deck for ASSEM
*P,F
    Relocatable decks for PASS 1
*T
*K,I8
*N,PASS1,,,B
*K,I6
    ...
    ...Same for PASSES 2-4
    ...
*NMACSKL,,,B
    Absolute MACSKL for library macros
*N,MACROS,,,B
    Absolute MACROS for library macros
*U
```

7.2.2 OPTIONAL TAPES

Tape 2

Tape 2 contains LIBRARY macro preparation programs and relocatable binary of LIBMAC, LIBMC2, and LIBMC3.

Tape 3

Tape 3 is the Macro Assembler source tape and contains the source of the LIBRARY macros in the following order:

FREAD

FWRITE

Q8A

Q8B

STATUS

READ

WRITE

INDIR

EXIT

CORE

LOADER

SCHDLE

TIMER

GTFILE

SPACE

RELEAS

Tape 5

Tape 5 is an optional magnetic tape containing in COSY format the sources for ASSEM, passes 1 through 4, and LIBMAC.

COSY Deck ID	Program	COSY Deck Name
AS	ASSEM	ASSEM
OO	PASS1	PASS1
OW	PA1PR2	PA1PR2
WO	PASS2	PASS2
WW	PA2PR2	PA2PR2
TO	PASS3	PASS3
TW	PA3PR2	PA3PR2
TT	PA3PR3	PA3PR3
FO	PASS4	PASS4
LB1	LIBMAC	LIBMAC
LB2	LIBMC2	LIBMC2
LB3	LIBMC3	LIBMC3

Tape 6

Tape 6 is a Hollerith listing of COSY programs.

7.3 COSY 1.0

The installation tape has the following format

COSY - RELOCATABLE BINARY

EOF

EOF

The COSY source tape has the following format

DECK - COSY

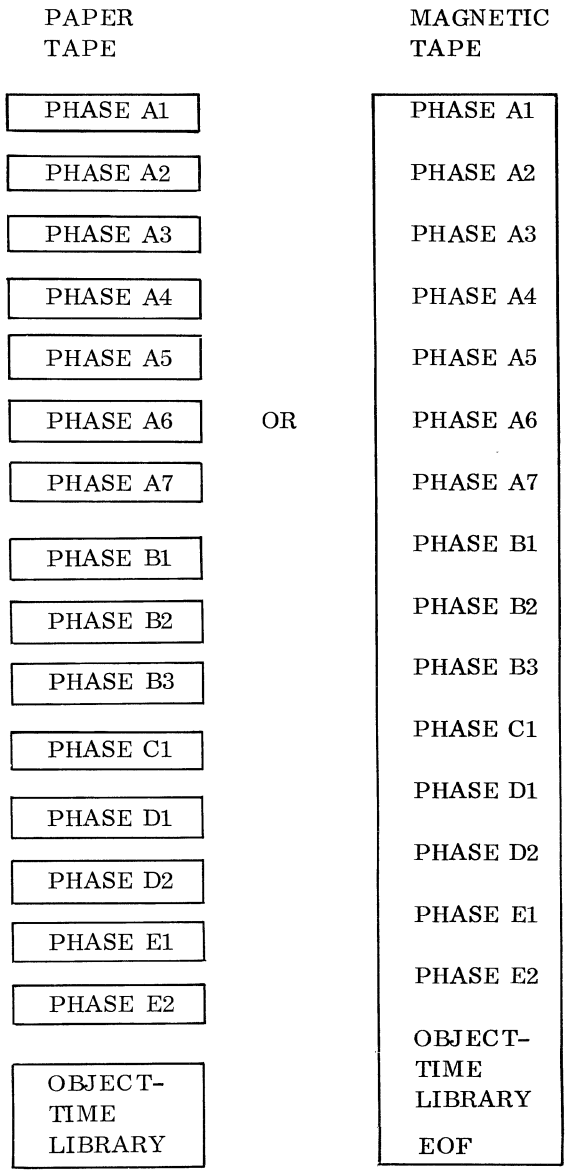
EOF

7.4 MASS STORAGE FORTRAN 2.0A

The installation material is on one magnetic tape or on 16 paper tapes with each phase on a different paper tape. Source information is on one magnetic tape in COSY format. SELCOP and IOCAL are on one paper tape. There are three list magnetic tapes.

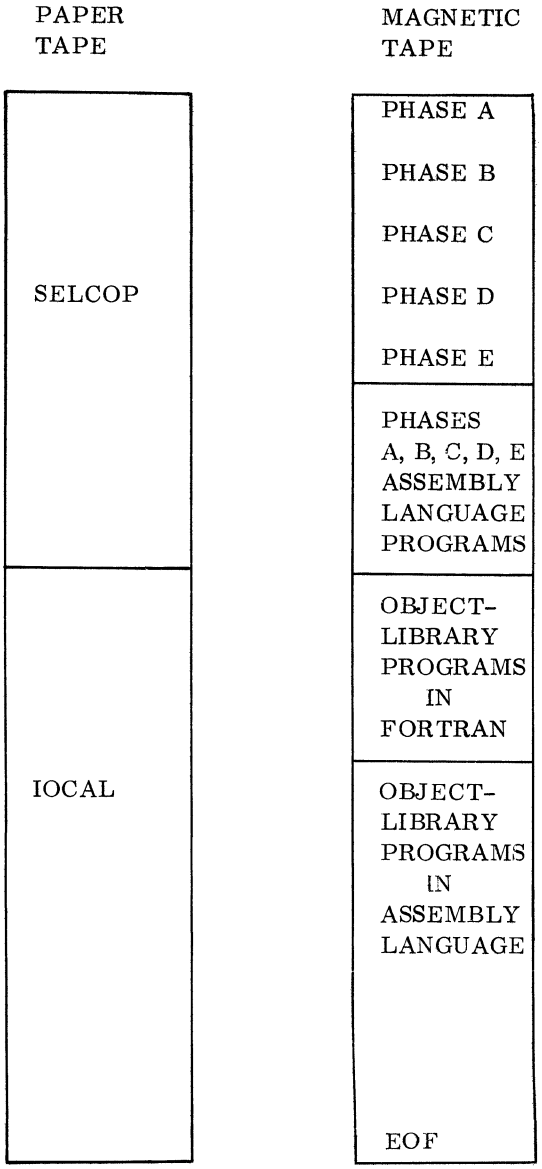
Installation Tapes

For further information on the installation tapes, see section 7.4.1.



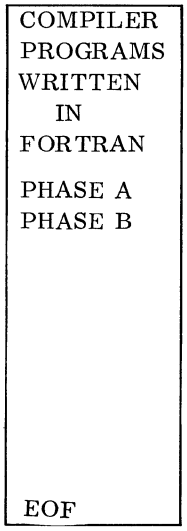
COSY Source

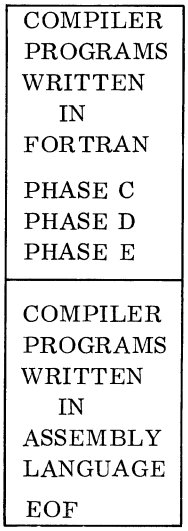
Further information on the COSY magnetic tape is in section 7.4.2.

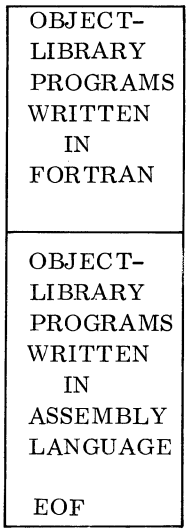


List Magnetic Tapes

For further information on the contents of these tapes, see sections 7.4.3 (compiler program order), 7.4.4 (compiler program lengths, common lengths, and externals), and 7.4.5 (object library program entry points and externals).







7.4.1 INSTALLATION TAPES

The installation tape has the following format for magnetic tape. For paper tape, *K,I6,P8 is replaced by *K,I2,P8 and there is an *U at the end of each physical tape.

```
*K,I6,P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA1
          DUMYA1
          ENDDO
          GETC
          GETF
          GNST
          IGETCF
          OPTION
          OUTENT
          PHASEA
          PLABEL
          Q8QBDS
          RDLABL
          STCHAR
          TYPE
          ENDLOC
*T
*K,I8
*N,FORTA1,,,B
*K,I6,P8
*P
          FTN
          GOA
          CNVT
          CONV
```

```
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA2
          DUMYA2
          ARITH
          COMNPR
          DIMPR
          GETC
          GETF
          SUBSCR
          TYPEPR
          ENDLOC
*T
*K,I8
*N,FORTA2,,,B
*K,I6,P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA3
          DUMYA3
          BYEQPR
          CHECKF
          CONSUB
          DATAPR
          FGETC
          FORK
          GETC
```

```
          GETF
          STCHAR
          TREE
          ENDLOC
*T
*K, I8
*N, FORTA3, , , B
*K, I6, P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA4
          DUMYA4
          ARAYSZ
          ASGNPR
          BDOPR
          CFIVOC
          CKIVC
          CKNAME
          CPLOOP
          ENDDO
          GETC
          GETF
          IOSPR
          OUTENT
          RDLABL
          STCHAR
          ENDLOC
*T
*K, I8
*N, FORTA4, , , B
*K, I6, P8
*P
          FTN
          GOA
          CNVT
          CONV
```

```
            DIAG
            EXP9
            FLOAT
            GETSYM
            GPUT
            IOPRBA
            PACK
            Q8PRMS
            STORE
            SYMBOL
            LOCLA5
            DUMYA5
            ARITH
            GETC
            GETF
            SUBSCR
            ENDLOC
*T
*K,I8
*N, FORTA5,,,B
*K,I6,P8
*P
            FTN
            GOA
            CNVT
            CONV
            DIAG
            EXP9
            FLOAT
            GETSYM
            GPUT
            IOPRBA
            PACK
            Q8PRMS
            STORE
            SYMBOL
            LOCLA6
            DUMYA6
            CFIVOC
            CKIVC
            ERBPR
            GETC
            GETF
            MODMXR
            RDLABL
            SUBPPR
            TREE
            ENDLOC
```

```
*T
*K,I8
*N,FORTA6,,,B
*K,I6,P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA7
          DUMYA7
          ASEMPR
          EXRLPR
          GETC
          GETF
          IGETCF
          PEQVS
          PRNTNM
          PUNT
          RDLABL
          SYMSCN
          ENDLOC
*T
*K,I8
*N,FORTA7,,,B
*K,I6,P8
*P
          FTN
          GOB
          CNVT
          DUMMY
          FCMSTK
          GETSYM
          IOPRBB
          KCPART
          KOUTPT
          KPCSTK
          KPC3PR
          KSYMGN
```

```
            LABKPC
            LABLER
            PUNT
            Q8PRMS
            STORE
            SYMBOL
            TSALOC
            LOCLB1
            DUMYB1
            ARAYSZ
            ASSEM
            BANANA
            BGINDO
            END
            ENTCOD
            HELEN
            INXRST
            NOPROC
            PHASEB
            READIR
            SUBFUN
            SYMSCN
            ENDLOC
*T
*K,I8
*N,FORTB1,,,B
*K,I6,P8
*P
            FTN
            GOB
            CNVT
            DUMMY
            FCMSTK
            GETSYM
            IOPRBB
            KCPART
            KOUTPT
            KPCSTK
            KPC3PR
            KSYMGN
            LABKPC
            LABLER
            PUNT
            Q8PRMS
            STORE
            SYMBOL
            TSALOC
            LOCLB2
```

```
            ACP
            AFIDL
            ASUPER
            CGOTO
            FINK
            INTRAM
            PARTSB
            SUBPR1
            SUBPR2
            SUBPR3
            ENDLOC
*T
*K,I8
*N,FORTB2,,,B
*K,I6,P8
*P
            FTN
            GOB
            CNVT
            DUMMY
            FCMSTK
            GETSYM
            IOPRBB
            KCPART
            KOUTPT
            KPCSTK
            KPC3PR
            KSYMGN
            LABKPC
            LABLER
            PUNT
            Q8PRMS
            STORE
            SYMBOL
            TSALOC
            LOCLB3
            ACP
            ARITHR
            ASUPER
            FINK
            INTRAM
            PARTSB
            SUBPR1
            SUBPR2
            SUBPR3
            ENDLOC
```

```
*T
*K,I8
*N,FORTB3,,,B
*K,I6,P8
*P
          FTN
          GOC
          BKDWN
          BLDUP
          BSS
          CHKWD
          CHOP
          CL12
          CON
          COUNT
          DATAST
          GETSYM
          INOUT
          IXOPT
          PHASEC
          LABEL
          LABIN
          QXLD
          REED
          SKIP
          SYMSCN
          IOPRBC
          Q8PRMS
          ENDLOC
*T
*K,I8
*N,FORTC1,,,B
*K,I6,P8
*P
          FTN
          GOOD
          INDEX
          IOPRBD
          NPUNCH
          Q8PRMS
          PHASE6
          LOCLD1
          DUMYD1
          AMT
          AMOUT
          ADMAX
          BKDWN
          COUNT
          LABOUT
```

```
        NP2OUT
        RBDX
        RBPK
        TABDEC
        UNPUNC
        GETSYM
        SYMSCN
        ENDLOC
*T
*K,I8
*N,FORTD1,,,B
*K,I6,P8
*P
        FTN
        GOOD
        INDEX
        IOPRBD
        NPUNCH
        Q8PRMS
        PHASE6
        LOCLD2
        DUMYD2
        AMT
        GETSYM
        IACON
        IHCON
        NWRITE
        PACK
        SYMSCN
        BEGINO
        FINISH
        ENDLOC
*T
*K,I8
*N,FORTD2,,,B
*K,I6,P8
*P
        FTN
        GOE
        INDEX
        IOPRBD
        NPUNCH
        Q8PRMS
        PHASE6
        LOCLD1
        DUMYD1
        AMT
        AMOUT
        ADMAX
```

```
            BKDWN
            COUNT
            LABOUT
            NP2OUT
            RBDX
            RBPK
            TABDEC
            UNPUNC
            CONV
            GETSYM
            IACON
            IHCON
            NWRITE
            PACK
            SETPRT
            SYMSCN
            ENDLOC
*T
*K,I8
*N,FORTE1,,,B
*K,I6,P8
*P
            FTN
            GOE
            INDEX
            IOPRBD
            NPUNCH
            Q8PRMS
            PHASE6
            LOCLD2
            DUMYD2
            AMT
            CONV
            GETSYM
            IACON
            IHCON
            NWRITE
            PACK
            SETPRT
            SYMSCN
            BEGINO
            FINISH
            ENDLOC
*T
*K,I8
*N,FORTE2,,,B
*K,I6,P8
*L,FTN
             FTN
```

```
*L,Q8IFRM
          Q8IFRM
*L,Q8FS
          Q8FS
*L,Q8TRAN
          Q8TRAN
*L,FLOT
          FLOAT
*L,Q8QINI
          Q8QINI
*L,Q8QEND
          Q8QEND
*L,Q8CMP1
          Q8CMP
*L,Q8RWBU
          Q8RWBU
*L,Q8ERRM
          Q8ERRM
*L,Q8DFNF
          Q8DFIO
*L, Q8QX
          Q8QX
*L,Q8QUNI
          Q8QUNI
*L, Q8FGET
          Q8FGET
*L,Q8MAGT
          Q8MAGT
*L,Q8GBCK
          TAPCON
*L,IOCK
          IOCK
*L,Q8PSE
          PSSTOP
*L,Q8PAND
          Q8PAND
*L,Q8EXP9
          Q8EXP9
*L,Q8EXP1
          Q8EXP1
*L,Q8AB
          Q8AB
*L,SIGN
          SIGN
*L,EXP
          EXPPRG
*L,SQRT
          SQRTF
```

```
*L, ALOG
        LNUPRG
*L, TANH
        TANH
*L, SIN
        SINCOS
*L, ATAN
        ARCTPG
*L, QSAVE
        Q8EXPN
*L, IFALT
        IFALT
*L, Q8FX
        FXFL
*L, Q8PREP
        Q8PRMS
*U
```

7.4.2 COSY SOURCE TAPE

The FORTRAN 2.0A source tape is in COSY (compressed) format. The programs are arranged in the following order:

1. Phase A programs written in FORTRAN.
2. Phase B programs written in FORTRAN.
3. Phases C, D, E programs written in FORTRAN.
4. Phase A, B, C, D, E programs written in assembly language.
5. Object library programs written in FORTRAN.
6. Object library programs written in assembly language.

To assemble or to compile a program, convert from COSY format into hollerith format and then work with the hollerith tape. Following is a list of sequence numbers and COSY deck names of the FORTRAN routine names.

Phase A Programs Written in FORTRAN

COSY Deck Identifier	Program	COSY Deck Name	Phases
A1	CNVT	CNVT	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3
A2	GPUT	GPUT	A1, A2, A3, A4, A5, A6, A7
A3	SYMBOL	SYMBL1	A1, A2, A3, A4, A5, A6, A7
A4	†GETF	GETF1	A1, A2, A3, A4, A6, A7

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
A5	GNST	GNST	A1
A6	OUTENT	OUTENT	A1, A4
A7	PHASEA	PHASEA	A1
A8	PLABEL	PLABEL	A1
A9	Q8QBDS	Q8QBDS	A1
A10	RDLABL	RDLABL	A1, A4, A6, A7
A11	STCHAR	STCHAR	A1, A3, A4
A12	TYPE	TYPE	A1
A13	†ARITH	ARITH1	A2
A14	COMNPR	COMNPR	A2
A15	DIMPR	DIMPR	A2
A16	SUBSCR	SUBSCR	A2, A5
A17	TYPEPR	TYPEPR	A2
A18	BYEQPR	BYEQPR	A3
A19	CHECKF	CHECKF	A3
A20	FGETC	FGETC	A3
A21	FORK	FORK	A3
A22	†ARITH	ARITH2	A5
A23	†GETF	GETF2	A5
A24	SUBPPR	SUBPPR	A6
A25	EXRLPR	EXRLPR	A7
A26	PEQVS	PEQVS	A7
A27	PRNTNM	PRNTNM	A7
A28	*PUNT	PUNT1	A7
A29	*SYMSCN	SMSCN4	A7, B1
A30	ENDDO	ENDDO	A1, A4
A31	CONSUB	CONSUB	A3
A32	DATAPR	DATAPR	A3
A33	ASGNPR	ASGNPR	A4
A34	BDOPR	BDOPR	A4

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
A35	CFIVOC	CFIVOC	A4, A6
A36	CKIVC	CKIVC	A4, A6
A37	CKNAME	CKNAME	A4
A38	IOSPR	IOSPR	A4
A39	ERBPR	ERBPR	A6
A40	MODMXR	MODMXR	A6
A41	ASEMPR	ASEMPR	A7
A42	TREE	TREE	A3, A6
A43	ARAYSZ	ARAYSZ	A4, B1
A44	CPLOOP	CPLOOP	A4

Phase B Programs Written in FORTRAN

COSY Deck Identifier	Program	COSY Deck Name	Phases
A50	DUMMY	DUMMY	B1, B2, B3
A51	FCMSTK	FCMSTK	B1, B2, B3
A52	KCPART	KCPART	B1, B2, B3
A53	KOUTPT	KOUTPT	B1, B2, B3
A54	KPCSTK	KPCSTK	B1, B2, B3
A55	KPC3PR	KPC3PR	B1, B2, B3
A56	KSYMGN	KSYMGN	B1, B2, B3
A57	LABKPC	LABKPC	B1, B2, B3
A58	LABLER	LABLER	B1, B2, B3
A59	†PUNT	PUNT2	B1, B2, B3
A60	†SYMBOL	SYMBL2	B1, B2, B3
A61	TSALOC	TSALOC	B1, B2, B3
A62	ASSEM	ASSEM	B1
A63	BANANA	BANANA	B1
A64	BGINDO	BGINDO	B1
A65	END	END	B1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
A66	ENTCOD	ENTCOD	B1
A67	HELEN	HELEN	B1
A68	INXRST	INXRST	B1
A69	NOPROC	NOPROC	B1
A70	PHASEB	PHASEB	B1
A71	READIR	READIR	B1
A72	SUBFUN	SUBFUN	B1
A73	ACP	ACP	B2, B3
A74	AFIDL	AFIDL	B2
A75	ASUPER	ASUPER	B2, B3
A76	CGOTO	CGOTO	B2
A77	FINK	FINK	B2, B3
A78	INTRAM	INTRAM	B2, B3
A79	PARTSB	PARTSB	B2, B3
A80	SUBPR1	SUBPR1	B2, B3
A81	SUBPR2	SUBPR2	B2, B3
A82	SUBPR3	SUBPR3	B2, B3
A83	ARITHR	ARITHR	B3

Phases C, D, and E Programs Written in FORTRAN

COSY Deck Identifier	Program	COSY Deck Name	Phases
B01	†BKDWN	BKDWN1	C
B02	BLDUP	BLDUP	C
B03	BSS	BSS	C
B04	CHKWD	CHKWD	C
B05	CHOP	CHOP	C
B06	CL12	CL12	C
B07	CON	CON	C
B08	†COUNT	COUNT1	C

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
B09	DATAST	DATAST	C
B10	†GETSYM	GTSYM1	C
B11	INOUT	INOUT	C
B12	IXOPT	IXOPT	C
B13	PHASEC	PHASEC	C
B14	LABEL	LABEL	C
B15	LABIN	LABIN	C
B16	QXLD	QXLD	C
B17	REED	REED	C
B18	SKIP	SKIP	C
B19	†SYMSCN	SMSCN1	C
B20	†INDEX	INDEX1	D1, D2
B21	†NPUNCH	NPNCH1	D1, D2
B22	†PHASE6	PHSE61	D1, D2
B23	†AMT	AMT1	D1, E1
B24	†AMOUT	AMOUT1	D1
B25	†ADMAX	ADMAX1	D1
B26	†BKDWN	BKDWN2	D1
B27	†COUNT	COUNT2	D1
B28	†LABOUT	LBOUT1	D1
B29	†NP2OUT	NP2OT1	D1
B30	†RBDX	RBDX1	D1
B31	†RBPK	RBPK1	D1
B32	†TABDEC	TBDEC1	D1
B33	†UNPUNC	UNPNC1	D1
B34	†GETSYM	GTSYM2	D1
B35	†SYMSCN	SMSCN2	D1
B36	†AMT	AMT2	D2, E2
B37	†GETSYM	GTSYM3	D2
B38	†IACON	IACON1	D2

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
B39	†IHCON	IHCON1	D2
B40	†NWRITE	NWRTE1	D2
B41	†SYMSCN	SMSCN3	D2, E1, E2
B42	†BEGINO	BEGNO1	D2
B43	†FINISH	FNISH1	D2
B44	†INDEX	INDEX2	E1, E2
B45	†NPUNCH	NPNCH2	E1, E2
B46	†PHASE6	PHSE62	E1, E2
B47	†AMOUT	AMOUT2	E1
B48	†ADMAX	ADMAX2	E1
B49	†BKDWN	BKDWN3	E1
B50	†COUNT	COUNT3	E1
B51	†LABOUT	LBOUT2	E1
B52	†NP2OUT	NP2OT2	E1
B53	†RBDX	RBDX2	E1
B54	†RBPK	RBPK2	E1
B55	†TABDEC	TBDEC2	E1
B56	†UNPUNC	UNPNC2	E1
B57	†GETSYM	GTSYM4	E1, E2
B58	†IACON	IACON2	E1, E2
B59	†IHCON	IHCON2	E1, E2
B60	†NWRITE	NWRTE2	E1, E2
B61	SETPRT	SETPRT	E1, E2
B62	†BEGINO	BEGNO2	E2
B63	†FINISH	FNISH2	E2

Compiler Programs Written in Assembly language

COSY Deck Identifier	Program	COSY Deck Name	Phases
C01	FTN	FTN4	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C, D1, D2, E1, E2

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
C02	GOA	GOA	A1, A2, A3, A4, A5, A6, A7
C03	†CONV	CONV1	A1, A2, A3, A4, A5, A6, A7
C04	DIAG	DIAG	A1, A2, A3, A4, A5, A6, A7
C05	EXP9	EXP9	A1, A2, A3, A4, A5, A6, A7
C06	†FLOAT	FLOAT1	A1, A2, A3, A4, A5, A6, A7
C07	†GETSYM	GTSYM5	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3
C08	IOPRBA	IOPRBA	A1, A2, A3, A4, A5, A6, A7
C09	PACK	PACK	A1, A2, A3, A4, A5, A6, A7, D2, E1, E2
C10	Q8PRMS	Q8PRMS	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C, D1, D2, E1, E2, Object Library
C11	STORE	STORE	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3
C12	LOCLA1	LOCLA1	A1
C13	DUMYA1	DUMYA1	A1
C14	†GETC	GETC1	A1, A2, A3, A4, A6, A7
C15	IGETCF	IGETCF	A1, A7
C16	OPTIONS	OPTION	A1
C17	ENDLOC	ENDLOC	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C, D1, D2, E1, E2
C18	LOCLA2	LOCLA2	A2
C19	DUMYA2	DUMYA2	A2
C20	LOCLA3	LOCLA3	A3
C21	DUMYA3	DUMYA3	A3
C22	LOCLA4	LOCLA4	A4
C23	DUMYA4	DUMYA4	A4
C24	LOCLA5	LOCLA5	A5
C25	DUMYA5	DUMYA5	A5
C26	†GETC	GETC2	A5
C27	LOCLA6	LOCLA6	A6
C28	DUMYA6	DUMYA6	A6
C29	LOCLA7	LOCLA7	A7
C30	DUMYA7	DUMYA7	A7

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
C31	GOB	GOB	B1, B2, B3
C32	IOPRBB	IOPRBB	B1, B2, B3
C33	LOCLB1	LOCLB1	B1
C34	DUMYB1	DUMYB1	B1
C35	LOCLB2	LOCLB2	B2
C36	LOCLB3	LOCLB3	B3
C37	GOC	GOC	C
C38	IOPRBC	IOPRBC	C
C39	GOOD	GOOD	D1, D2
C40	IOPRBD	IOPRBD	D1, D2, E1, E2
C41	LOCLD1	LOCLD1	D1, E1
C42	DUMYD1	DUMYD1	D1, E1
C43	LOCLD2	LOCLD2	D2, E2
C44	DUMYD2	DUMYD2	D2, E2
C45	GOE	GOE	E1, E2
C46	†CONV	CONV2	E1, E2

Object Library Programs Written in FORTRAN

COSY Deck Identifier	Program	COSY Deck Name
Q01	Q8IFRM	Q8IFRM
Q02	Q8FS	Q8FS
Q03	Q8TRAN	Q8TRAN

Object Library Programs written in Assembly language

COSY Deck Identifier	Program	COSY Deck Name
Q04	†FLOAT	FLOAT2
Q05	Q8QINI	Q8QINI

† Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name
Q06	Q8QEND	Q8QEND
Q07	Q8CMP	Q8CMP
Q08	Q8RWBU	Q8RWBU
Q09	Q8ERRM	Q8ERRM
Q10	Q8DFIO	Q8DFIO
Q11	Q8QX	Q8QX
Q12	Q8QUNI	Q8QUNI
Q13	Q8FGET	Q8FGET
Q14	Q8MAGT	Q8MAGT
Q15	TAPCON	TAPCON
Q16	IOCK	IOCK
Q17	PSSTOP	PSSTOP
Q18	Q8PAND	Q8PAND
Q19	Q8EXP9	Q8EXP9
Q20	Q8EXP1	Q8EXP1
Q21	Q8AB	Q8AB
Q22	SIGN	SIGN
Q23	EXPPRG	EXPPRG
Q24	SQRTF	SQRTF
Q25	LNUPRG	LNUPRG
Q26	TANH	TANH
Q27	SINCOS	SINCOS
Q28	ARCTPG	ARCTPG
Q29	Q8EXPN	Q8EXPN
Q30	IFALT	IFALT
Q31	FXFL	FXFL

7.4.3 COMPILER PROGRAM ORDER

The compiler consists of four passes over the source code or its equivalent, accomplished in four phases called A, B, C, and D/E. (The fourth pass is performed by either Phase D or Phase E depending upon whether the user wants to use an assembly language listing output.) Each phase consists of a root which is core-resident throughout the phase, and zero or more local subroutine groups which share the same core area and are read from disc as needed. Phase A reads the source input, converts it to statements expressed in an internal code, and assigns a statement number to the statement.

Phase B reads the output of Phase A and generates pseudo code from it. (Pseudo code is similar to assembler input except that the index to be used in an indexed instruction and the addressing mode are not specified.)

Phase C and D/E are a two-pass assembler. The output from Phase B is read. Index registers are optimally assigned. One word relative addressing is maximized. Relocatable binary output and an assembly listing are produced.

Files	Root	Local
FORTA1	A	1
FORTA2	A	2
FORTA3	A	3
FORTA4	A	4
FORTA5	A	5
FORTA6	A	6
FORTA7	A	7
FORTB1	B	1
FORTB2	B	2
FORTB3	B	3
FORTC1	C	---
FORTD1	D	1
FORTD2	D	2
FORTE1	E	1

Phase A Programs

Root Programs:

FTN
GOA
CNVT
CONV

DIAG
EXP9
FLOAT
GETSYM
GPUT
IOPRBA
PACK
Q8PRMS
STORE
SYMBOL

Local 1 Programs:

LOCLA1
DUMYA1
ENDDO
GETC
GETF
GNST
IGETCF
OPTION
OUTENT
PHASEA
PLABEL
Q8QBDS
RDLABL
STCHAR
TYPE
ENDLOC

Local 2 Programs:

LOCLA2
DUMYA2
ARITH
COMNPR
DIMPR
GETC
GETF
SUBSCR
TYPEPR
ENDLOC

Local 3 Programs:

LOCLA3
DUMYA3
BYEQPR
CHECKF

CONSUB
DATAPR
FGETC
FORK
GETC
GETF
STCHAR
TREE
ENDLOC

Local 4 Programs:

LOCLA4
DUMYA4
ARAYSZ
ASGNPR
BDOPR
CFIVOC
CKIVC
CKNAME
CPLOOP
ENDDO
GETC
GETF
IOSPR
OUTENT
RDLABL
STCHAR
ENDLOC

Local 5 Programs:

LOCLA5
DUMYA5
ARITH
GETC
GETF
SUBSCR
ENDLOC

Local 6 Programs:

LOCLA6
DUMYA6
CFIVOC
CKIVC
ERBPR
GETC
GETF

MODMXR
RDLABL
SUBPPR
TREE
ENDLOC

Local 7 Programs:

LOCLA7
DUMYA7
ASEMPR
EXRLPR
GETC
GETF
IGETCF
PEQVS
PRNTNM
PUNT
RDLABL
SYMSCN
ENDLOC

Pass B Programs

Root Programs:

FTN
GOB
CNVT
DUMMY
FCMSTK
GETSYM
IOPRBB
KCPART
KOUTPT
KPCSTK
KPC3PR
KSYMGN
LABKPC
LABLER
PUNT
Q8PRMS
STORE
SYMBOL
TSALOC

Local 1 Programs:

LOCLB1
DUMYB1
ARAYSZ
ASSEM
BANANA
BGINDO
END
ENTCOD
HELEN
INXRST
NOPROC
PHASEB
READIR
SUBFUN
SYMSCN
ENDLOC

Local 2 Programs:

LOCLB2
ACP
AFIDL
ASUPER
CGOTO
FINK
INTRAM
PARTSB
SUBPR1
SUBPR2
SUBPR3
ENDLOC

Local 3 Programs:

LOCLB3
ACP
ARITHR
ASUPER
FINK
INTRAM
PARTSB
SUBPR1
SUBPR2
SUBPR3
ENDLOC

Pass C Programs

FTN
GOC
BKDWN
BLDUP
BSS
CHKWD
CHOP
CL12
CON
COUNT
DATAST
GETSYM
INOUT
IXOPT
PHASEC
LABEL
LABIN
QXLD
REED
SKIP
SYMSCN
IOPRBC
Q8PRMS
ENDLOC

Pass D Programs

Root Programs:

FTN
GOOD
INDEX
IOPRBD
NPUNCH
Q8PRMS
PHASE6

Local 1 Programs:

LOCLD1
DUMYD1
AMT
AMOUT
ADMAX
BKDWN
COUNT
LABOUT

NP2OUT
RBDX
RBPK
TABDEC
UNPUNC
GETSYM
SYMSCN
ENDLOC

Local 2 Programs:

LOCLD2
DUMYD2
AMT
GETSYM
IACON
IHCON
NWRITE
PACK
SYMSCN
BEGINO
FINISH
ENDLOC

Pass E Programs

Root Programs:

FTN
GOE
INDEX
IOPRBD
NPUNCH
Q8PRMS
PHASE6

Local 1 Programs:

LOCLD1
DUMYD1
AMT
AMOUT
ADMAX
BKDWN
COUNT
LABOUT
NP2OUT
RBDX
RBPK

TABDEC
UNPUNC
CONV
GETSYM
IACON
IHCON
NWRITE
PACK
SETPRT
SYMSCN
ENDLOC

Local 2 Programs:

LOCLD2
DUMYD2
AMT
CONV
GETSYM
IACON
IHCON
NWRITE
PACK
SETPRT
SYMSCN
BEGINO
FINISH
ENDLOC

7.4.4 COMPILER PROGRAM LENGTHS, COMMON LENGTHS, AND EXTERNALS

FORTRAN 2.0A Phase A Compiler Programs

Program Name	Common Labeled	Common Blank	Program Length	Externals
CNVT	1649	3	62	------
GPUT	1649	1236	41	------
SYMBOL	1649	3	162	GETSYM WRITE SKIPIT
GETF	1649	1236	724	GETC GPUT DIAG EXP9 CNVT SYMBOL

Program Name	Common Labeled	Common Blank	Program Length	Externals
GNST	1649	1236	446	CONV PACK WRITE IGETCF READ STCHAR EXIT DIAG SKIPIT
QUTENT	1649	1236	52	Q8PRUP Q8PREP WRITE
PHASEA	1649	1236	1259	WRITE GNST PLABEL TYPE DIAG OUTENT PEQVS DIMPR COMNPR TYPEPR SUBPPR GETF STORE BYEQPR EXRLPR GETC DATAPR CHECKF ARITH GETSYM SYMBOL ASGNPR RDLABL ENDDO CPLOOP SKIPIT ERBPR IOSPR BDOPR STCHAR ASEMPR

Program Name	Common Labeled	Common Blank	Program Length	Externals
PLABEL	1649	1236	86	RDLABL STORE DIAG
Q8QBDS	1649	----	0	------
RDLABL	1649	1236	129	GETC CNVT SYMBOL
STCHAR	----	----	50	------
TYPE	1649	1236	510	Q8PRUP Q8PREP GETC GETF
ARITH	1649	1236	1637	GETF DIAG PUNT TREE GETSYM SUBSCR STORE GETC SYMBOL
COMNPR	1649	1236	150	GETF DIMPR DIAG
DIMPR	1649	1236	391	GETF DIAG STORE GETSYM
SUBSCR	1649	1236	701	GETF DIAG STORE PUNT GETSYM SYMBOL
TYPEPR	1649	1236	23	DIMPR
BYEQPR	1649	1236	499	GETF DIAG STORE PUNT GETSYM

Program Name	Common Labeled	Common Blank	Program Length	Externals
CHECKF	1649	1236	160	FORK DIAG FGETC
FGETC	1649	1236	31	Q8PKUP Q8PREP GETC STCHAR
FORK	----	----	370	FGETC
ARITH	1649	1236	1630	GETF DIAG PUNT TREE GETSYM SUBSCR STORE GETC SYMBOL
GETF	1649	1236	724	GETC GPUT DIAG EXP9 CNVT SYMBOL
SUBPPR	1649	1236	192	GETF DIAG STORE
EXPLPR	1649	1236	94	GETF DIAG STORE
PEQVS	1649	1236	991	SYMSCN GETSYM PRNTNM DIAG
PRNTNM	1649	----	141	Q8PKUP Q8PREP GETSYM WRITE
PUNT	----	1236	56	DIAG READ IGETCF SKIPIT

Program Name	Common Labeled	Common Blank	Program Length	Externals
SYMSCN	1553	----	28	GETSYM
ENDDO	1649	1236	261	GETSYM OUTENT DIAG
CONSUB	1649	1236	135	DIAG GETF
DATAPR	1649	1236	400	GETF DIAG STORE PUNT CONSUB GETSYM
ASGNPR	1649	1236	70	RDLABL DIAG SYMBOL STORE GETC CKNAME
BDOPR	1649	1236	314	RDLABL DIAG STORE CKNAME CKIVC GETC GETSYM
CFIVOC	1649	1236	94	GETF DIAG STORE
CKIVC	----	1236	16	CFIVOC DIAG
CKNAME	----	1236	16	CFIVOC DIAG
IOSPR	1649	1236	1679	GETF DIAG SYMBOL STORE CKIVC RDLABL GETC STCHAR ENDDO BDOPR

Program Name	Common Labeled	Common Blank	Program Length	Externals
				OUTENT ARITH
ERBPR	1649	1236	83	CKIVC DIAG
MODMXR	1649	1236	1116	CNVT SYMBOL STORE PUNT
ASEMPR	1649	1236	434	GETC GETF DIAG STORE GETSYM RDLABL
TREE	1649	1236	1289	PUNT DIAG GETSYM MODMXR
ARAYSZ	1649	----	106	Q8PKUP Q8PREP
CPLOOP	1649	550	165	GETSYM ARAYSZ

FORTRAN 2.0A Compiler Programs: Phase B

Program Name	Common Labeled	Common Blank	Program Length	Externals
DUMMY	1739	1158	271	Q8PKUP Q8PREP GETSYM KSYMGN
FCMSTK	1739	1158	137	Q8PKUP Q8PREP KOUTPT
KCPART	----	----	49	Q8PKUP Q8PREP
KOUTPT	1739	1158	18	WRITE
KPCSTK	1739	1158	955	Q8PKUP Q8PREP KCPART GETSYM

Program Name	Common Labeled	Common Blank	Program Length	Externals
				DUMMY KOUTPT FCMSTK
KPC3PR	----	----	24	Q8PREP Q8PKUP KPCSTK
KSYMGN	1739	1158	72	Q8PKUP Q8PREP CNVT SYMBOL STORE
LABKPC	1739	1158	20	Q8PKUP Q8PREP KPCSTK
LABLER	1649	----	30	Q8PKUP Q8PREP KSYMGN
PUNT	1649	----	22	WRITE SKIPIT
SYMBOL	1649	3	157	GETSYM WRITE SKIPIT
TSALOC	1739	1158	139	Q8PKUP Q8PREP PUNT KSYMGN
ASSEM	1739	1158	103	KPCSTK LABKPC
BANANA	1739	1158	195	KPC3PR GETSYM LABKPC
BGINDO	1739	1158	265	PUNT LABLER GETSYM KPC3PR LABKPC
END	1739	1158	72	LABKPC INXRST KPC3PR ENTCOD GETSYM

Program Name	Common Labeled	Common Blank	Program Length	Externals
ENTCOD	1739	1158	169	KPC3PR LABKPC
HELEN	1739	1158	343	LABKPC KPC3PR ARAYSZ GETSYM SYMSCN
INXRST	1739	1158	20	KPC3PR
NOPROC	1739	1158	49	LABLER KPC3PR KOUTPT WRITE LABKPC
PHASEB	1739	1165	1109	LABLER LABKPC KPC3PR HELEN KPCSTK READIR TSALOC SUBFUN NOPROC ARITHR GETSYM ASUPER INXRST ENTCOD CGOTO BGINDO BANANA AFIDL PUNT ASSEM END
READIR	1739	1158	88	Q8PKUP Q8PREP PUNT READ KPC3PR LABKPC
SUBFUN	1739	1158	103	GETSYM LABLER

Program Name	Common Labeled	Common Blank	Program Length	Externals
ACP	1739	1158	1097	Q8PKUP Q8PREP PUNT INTRAM GETSYM KPC3PR KPCSTK TSALOC PARTSB FINK
AFIDL	1739	1158	90	Q8PKUP Q8PREP ASUPER KPC3PR
ASUPER	1739	1158	182	Q8PKUP Q8PREP PUNT SUBPR1 GETSYM SUBPR2 ACP
CGOTO	1739	1165	91	LABLER ASUPER KPC3PR KPCSTK LABKPC
FINK	1739	1158	181	KPCSTK LABLER KPC3PR LABKPC
INTRAM	1739	1158	473	PUNT KPCSTK KPC3PR TSALOC GETSYM LABKPC
PARTSB	1739	1158	174	Q8PKUP Q8PREP KPCSTK GETSYM KPC3PR SYMBOL STORE

Program Name	Common Labeled	Common Blank	Program Length	Externals
SUBPR1	1739	1158	62	Q8PKUP Q8PREP SUBPR3 TSALOC INTRAM KPC3PR
SUBPR2	1739	1158	141	Q8PKUP Q8PREP SUBPR3 KPC3PR GETSYM KPCSTK LABLER
SUBPR3	1739	1158	71	Q8PKUP Q8PREP ACP PARTSB
ARITHR	1739	1165	447	GETSYM SUBPR1 KPCSTK ASUPER KPC3PR

FORTRAN 2.0 A Phase C Compiler Programs

Program Name	Common Labeled	Common Blank	Program Length	Externals
BKDWN	2993	1148	95	------
BLDUP	2993	1148	67	------
BSS	2993	1148	30	BLDUP
CHKWD	2993	1148	382	GETSYM
CHOP	2993	1148	544	GETSYM
CL12	2993	1148	185	GETSYM CHOP BLDUP
CON	2993	1148	55	BLDUP
COUNT	2993	1148	23	------
DATAST	2993	1148	167	GETSYM BLDUP INOUT

Program Name	Common Labeled	Common Blank	Program Length	Externals
GETSYM	2993	1148	164	WRITE READ
INOUT	2993	1148	111	REED BKDWN WRITE
IXOPT	2993	1148	315	CHKWD BKDWN QXLD GETSYM
PHASEC	2993	1148	926	WRITE READ SYMSCN REED DATAST GETSYM CHOP LABEL BLDUP INOUT BSS COUNT BKDWN CHKWD LABIN IXOPT CON CL12 QXLD SKIP
LABEL	2993	1148	34	BLDUP INOUT
LABIN	2993	1148	102	REED LABEL
QXLD	2993	1148	144	Q8PKUP Q8PREP CHOP BLDUP INOUT COUNT
REED	2993	1148	93	READ
SKIP	2993	1148	86	CHOP BLDUP

Program Name	Common Labeled	Common Blank	Program Length	Externals
				INOUT COUNT
SYMSCN	2993	1148	28	GETSYM

FORTRAN 2.0A Phase D Compiler Programs

Program Name	Common Labeled	Common Blank	Program Length	Externals
INDEX	1073	3095	28	Q8PKUP Q8PREP
NPUNCH	1073	3095	319	WRITE RESET READ
PHASE6	1073	3095	160	BEGIN READ INDEX AMT FINISH NPUNCH
AMT	----	----	9	AMOUT
AMOUT	1073	3095	1507	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC NPUNCH SYMSCN
ADMAX	1073	3095	513	INDEX GETSYM TABDEC
BKDWN	1073	3095	105	INDEX
COUNT	1073	3095	23	------
LABOUT	1073	3095	223	Q8PKUP Q8PREP GETSYM UNPUNC RBPK

Program Name	Common Labeled	Common Blank	Program Length	Externals
NP2OUT	1073	3095	47	RBPK COUNT WRITE
RBDX	1073	3095	60	------
RBPK	1073	3095	42	RBDX UNPUNC
TABDEC	1073	3095	132	SYMSCN
UNPUNC	1073	3095	22	RBDX NPUNCH
GETSYM	1073	3095	60	WRITE READ
SYMSCN	1073	3095	39	GETSYM
AMT	----	----	7	------
GETSYM	1073	3095	46	READ
IACON	1073	3095	88	------
IHCON	1073	3095	45	Q8PKUP Q8PREP
NWRITE	1073	3095	59	PACK WRITE
SYMSCN	1073	----	28	GETSYM
BEGINO	1073	3095	428	READ NWRITE IHCON GETSYM IACON NPUNCH SYMSCN WRITE
FINISH	1073	3095	410	NWRITE IHCON SYMSCN IACON NPUNCH

FORTRAN 2.0A Phase E Compiler Programs

Program Name	Common Labeled	Common Blank	Program Length	Externals
INDEX	1073	2090	28	Q8PKUP Q8PREP
NPUNCH	1073	2090	319	WRITE RESET READ
PHASE6	1073	2090	160	BEGINO READ INDEX AMT FINISH NPUNCH
AMOUT	1073	2090	1513	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC SETPRT CONV NPUNCH
ADMAX	1073	2090	513	INDEX GETSYM TABDEC
BKDWN	1073	2090	105	INDEX
COUNT	1073	2090	23	------
LABOUT	1073	2090	274	Q8PKUP Q8PREP GETSYM UNPUNC RBPK NWRITE IACON IHCON
NP2OUT	1073	2090	56	RBPK SETPRT COUNT WRITE

Program Name	Common Labeled	Common Blank	Program Length	Externals
RBDX	1073	2090	61	------
RBPK	1073	2090	42	RBDX UNPUNC
TABDEC	1073	2090	124	SYMSCN
UNPUNC	1073	2090	22	RBDX NPUNCH
GETSYM	1073	2090	77	WRITE READ
IACON	1073	2090	88	------
IHCON	1073	2090	44	Q8PKUP Q8PREP
NWRITE	1073	2090	59	PACK WRITE
SETPRT	1073	2090	379	IHCON IACON CONV NWRITE
BEGINO	1073	2090	329	READ NWRITE IHCON GETSYM IACON SETPRT NPUNCH SYMSCN
FINISH	1073	2090	367	NWRITE IHCON SYMSCN IACON NPUNCH

FORTRAN 2.0A Object Library Programs

Program Name	Common Labeled	Common Blank	Program Length	Externals
Q8IRFM	----	----	64	Q8PKUP Q8PREP Q8FS Q8TRAN

Program Name	Common Labeled	Common Blank	Program Length	Externals
Q8FS	----	----	464	Q8STP Q8PKUP Q8PREP Q8SKIP Q8FGET Q8FERM Q8RWBU Q8FPUT
Q8TRAN	----	----	1741	Q8STP Q8PKUP Q8PREP Q8FS Q8RWBU Q8MOVE Q8FERM Q8EXP9 Q8EXP1

7.4.5 OBJECT LIBRARY PROGRAM ENTRY POINTS AND EXTERNALS

	Program Name	Entry Points	Externals
1.	FTN	FTN	
2.	Q8QINI	Q8QINI Q8UNIT Q8SKIP	Q8ERRM Q8INTB Q8EREM Q8QTOM Q8CMPO Q8MAGT Q8QEND Q8IFRM Q8IGP Q8CMP1 Q8QUN2 Q8DFIN
3.	Q8QEND	Q8QEND	Q8CMP1 Q8QUN1 Q8QUN2 Q8UNIT Q8DFAD
4.	Q8CMP	Q8CMP0 Q8CMP1 Q8DFAD Q8QENS	Q8EREM Q8BEGB Q8LOCB Q8CLRB

	Program Name	Entry Points	Externals
			Q8RINT
			Q8EOTT
5.	Q8RWBU	Q8BINB	Q8CMP1
		Q8LOCB	Q8EREM
		Q8RWBU	
		Q8INTB	
		Q8BEGB	
		Q8CLRB	
		Q8RINT	
		Q8IBUF	
6.	Q8ERRM	Q8ERRM	Q8LOCF
		Q8FERM	Q8LOCB
		Q8EREM	
7.	Q8DFIO	Q8DFNF	Q8ERRM
		Q8DFIN	Q8EREM
			Q8QINI
			Q8DFAD
			Q8QENS
8.	Q8QX	Q8QTOM	Q8IFRM
		Q8QTRM	Q8BINB
		Q8QX	Q8QUN1
		Q8MOVE	Q8UNIT
		Q8QY	
9.	Q8QUNI	Q8QUN1	Q8EREM
		Q8QUN2	
		Q8QUN3	
10.	Q8FGET	Q8FGET	
		Q8FPUT	
		Q8IGP	
		Q8LOCF	
11.	Q8MAGT	Q8MAGT	Q8EREM
		Q8EOTT	Q8QUN2
			Q8COMI
			Q8QWND
12.	TAPCON	Q8QBCK	Q8EREM
		Q8QFLE	Q8CMP0
		Q8QWND	Q8CMP1
		EOF	Q8QUN1
			Q8QUN2
			Q8QUN3
			Q8IBUF
13.	IOCK	IOCK	Q8QUN1
			Q8QUN3

	Program Name	Entry Points	Externals
14.	PSSTOP	Q8PSE Q8PSEN Q8STP Q8STPN Q8COMI	Q8PAND
15.	Q8PAND	Q8PAND	
16.	Q8EXP9	Q8EXP9 Q8EXPT Q8EXP2	FLOT
17.	Q8EXP1	Q8EXP1	FLOT Q8EXPT Q8EXP2
18.	Q8IFRM	Q8IFRM	Q8FS Q8TRAN Q8PKUP Q8PREP
19.	Q8FS	Q8FS	Q8SKIP Q8FGET Q8FERM Q8RWBU Q8FPUT Q8STP Q8PKUP Q8PREP
20.	Q8TRAN	Q8TRAN	Q8FS Q8RWBU Q8FERM Q8EXP9 Q8EXP1 Q8STP Q8PKUP Q8PREP Q8MOVE
21.	Q8AB	Q8AB ABS	FLOT
22.	SIGN	Q8SG SIGN	FLOT
23.	EXPPRG	EXP	FLOT
24.	SQRTF	SQRT	FLOT
25.	LNUPRG	ALOG	FLOT
26.	TANH	TANH	EXP FLOT

	Program Name	Entry Points	Externals
27.	SINCOS	SIN COS	FLOT
28.	ARCTPG	ATAN	FLOT
29.	Q8EXPN	Q8QF2I Q8QI2F Q8QF2F RETAD QSAVE	FLOT ALOG EXP
30.	FLOAT	FLOT	
31.	Q8PRMS	Q8PREP Q8PKUP	
32.	IFALT	IFALT	
33.	FXFL	Q8QFIX Q8FX Q8FLT Q8FLOT IFIX FLOAT	FLOT

7.5 MASS STORAGE FORTRAN 2.0B

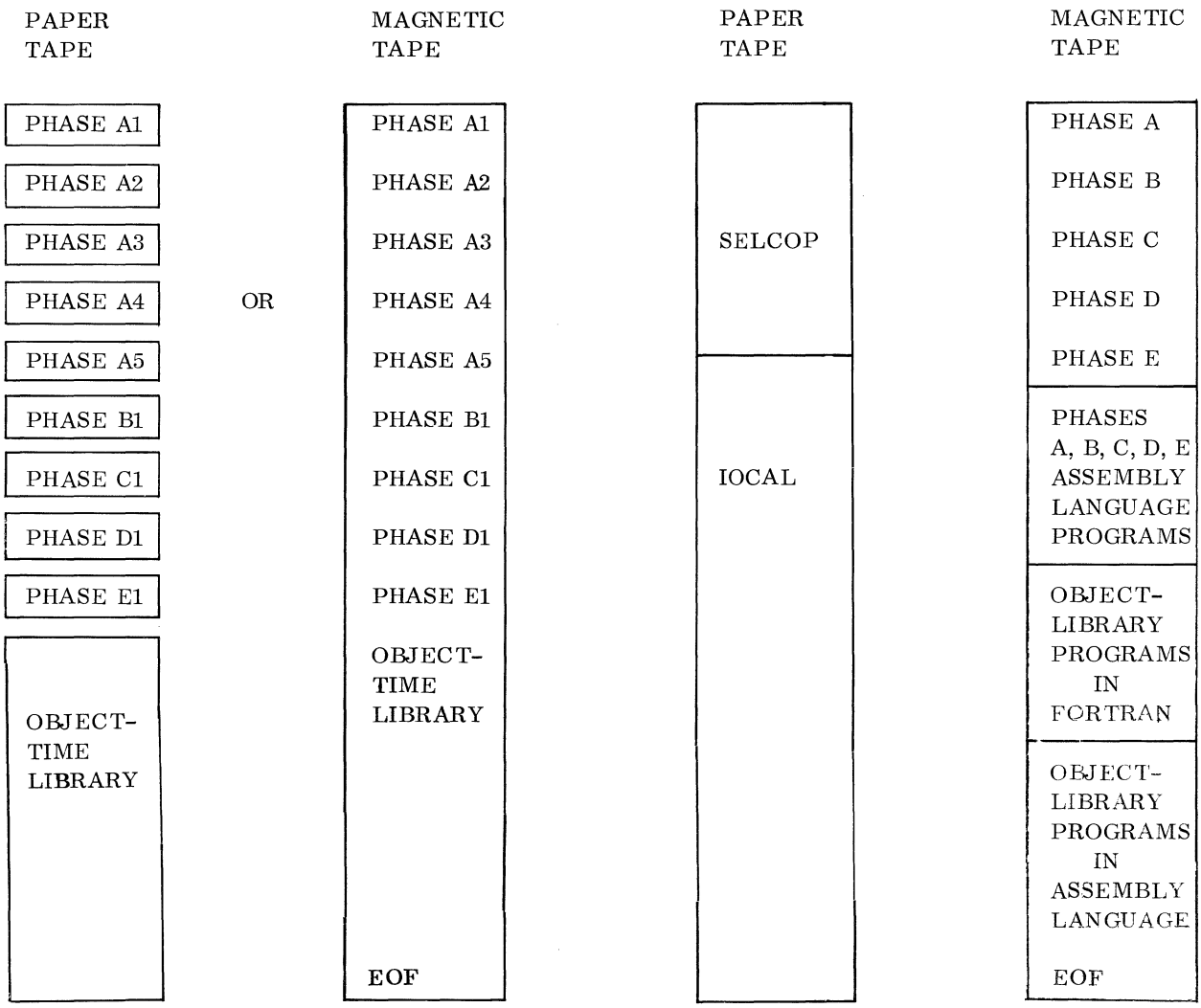
The installation material for FORTRAN 2.0B is on one magnetic tape and on ten paper tapes with each phase on a different paper tape. Source information is on one magnetic tape in COSY format. SELCOP and IOCAL are on one paper tape. There are three list magnetic tapes.

Installation Tapes

For further information on the installation tapes, see section 7.5.1.

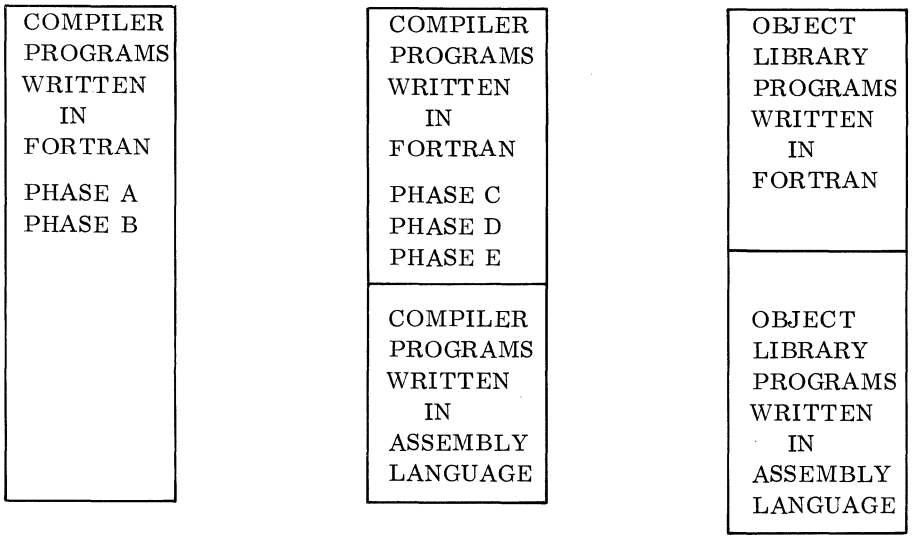
COSY Source

For further information on the COSY source magnetic tape, see section 7.5.2.



List Magnetic Tapes

For further information on the contents of these tapes, see sections 7.5.3 (compiler program order), 7.5.4 (compiler program lengths, common lengths, and externals), and 7.5.5 (object library program entry points and externals).



7.5.1 INSTALLATION TAPE

The installation tape has the following format for magnetic tape. For paper tape, *K,I6,P8 is replaced by *K,I2,P8 and there is a*U at the end of each physical tape.

```
*K,I6,P8
*P
        FTN
        GOA
        CFIVOC
        CKNAME
        CNVT
        CONV
        DIAG
        EXP9
        FLOAT
        GETC
        GETF
        GETSYM
        GPUT
        IGETCF
        IOPRBA
        PACK
```

```
            Q8PRMS
            RDLABL
            STORE
            SYMBOL
            ENDDO
            GNST
            OPTION
            OUTENT
            PHASEA
            PLABEL
            STCHAR
            TYPE
            LOCLA1
            DUMYA1
            Q8QBDS
            ENDLOC
*T
*K,I8
*N,FORTA1,,,B
*K,I6,P8
*P
            FTN
            GOA
            CFIVOC
            CKNAME
            CNVT
            CONV
            DIAG
            EXP9
            FLOAT
            GETC
            GETF
            GETSYM
            GPUT
            IGETCF
            IOPRBA
            PACK
            Q8PRMS
            RDLABL
            STORF
            SYMBOL
            ENDDO
            GNST
            OPTION
            OUTENT
            PHASEA
            PLABEL
            STCHAR
            TYPE
```

```
            LOCLA2
            DUMYA2
            BYEQPR
            CHECKF
            COMNPR
            CONSUB
            DATAPR
            DIMPR
            EXRLPR
            FGETC
            FORK
            PEQVS
            PRNTNM
            SUBPPR
            SYMSCN
            TYPEPR
            ENDLOC
*T
*K,I8
*N,FORTA2,,,B
*K,I6,P8
*P
            FTN
            GOA
            CFIVOC
            CKNAME
            CNVT
            CONV
            DIAG
            EXP9
            FLOAT
            GETC
            GETF
            GETSYM
            GPUT
            IGETCF
            IOPRBA
            PACK
            Q8PRMS
            RDLABL
            STORE
            SYMBOL
            ENDDO
            GNST
            OPTION
            OUTENT
            PHASEA
            PLABEL
            STCHAR
```

```
        TYPE
        LOCLA3
        DUMYA3
        ARAYSZ
        ASEMPR
        ASGNPR
        BDOPR
        CHECKF
        CKIVC
        CONSUB
        CPLOOP
        DATAPR
        FGETC
        FORK
        ERBPR
        MODMXR
        PUNT
        ENDLOC
*T
*K,I8
*N,FORTA3,,,B
*K,I6,P8
*P
        FTN
        GOA
        CFIVOC
        CKNAME
        CNVT
        CONV
        DIAG
        EXP9
        FLOAT
        GETC
        GETF
        GETSYM
        GPUT
        IGETCF
        IOPRBA
        PACK
        Q8PRMS
        RDLABL
        STORE
        SYMBOL
        ENDDO
        GNST
        OPTION
        OUTENT
        PHASEA
        PLABEL
```

```
          STCHAR
          TYPE
          LOCLA4
          DUMYA4
          ARITH
          SUBSCR
          TREE
          ENDLOC
*T
*K,I8
*N,FORTA4,,,B
*K,I6,P8
*P
          FTN
          GOA
          CFIVOC
          CKNAME
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETC
          GETF
          GETSYM
          GPUT
          IGETCF
          IOPRBA
          PACK
          Q8PRMS
          RDLABL
          STORE
          SYMBOL
          ENDDO
          GNST
          OPTION
          OUTENT
          PHASEA
          PLABEL
          STCHAR
          TYPE
          LOCLA5
          DUMYA5
          BDOPR
          CKIVC
          IOSPR
          ENDLOC
```

```
*T
*K,I8
*N,FORTA5,,,B
*K,I6,P8
*P
        FTN
        GOB
        CNVT
        DUMMY
        FCMSTK
        GETSYM
        IOPRBB
        KCPART
        KOUTPT
        KPCSTK
        KPC3PR
        KSYMGN
        LABKPC
        LABLER
        PUNT
        Q8PRMS
        STORE
        SYMBOL
        TSALOC
        ARAYSZ
        ASSEM
        BANANA
        BGINDO
        END
        ENTCOD
        HELEN
        INXRST
        NOPROC
        PHASEB
        READIR
        SUBFUN
        SYMSCN
        ACP
        AFIDL
        ASUPER
        CGOTO
        FINK
        INTRAM
        PARTSB
        SUBPR1
        SUBPR2
        SUBPR3
        ARITHR
        ENDLOC
```

```
*T
*K,I8
*N,FORTB1,,,B
*K,I6,P8
*P
          FTN
          GOC
          BKDWN
          BLDUP
          BSS
          CHKWD
          CHOP
          CL12
          CON
          COUNT
          DATAST
          GETSYM
          INOUT
          IOPRBC
          IXOPT
          LABEL
          LABIN
          PHASEC
          Q8PRMS
          QXLD
          REED
          SKIP
          SYMSCN
          ENDLOC
*T
*K,I8
*N,FORTC1,,,B
*K,I6,P8
*P
          FTN
          GOOD
          AMOUT
          ADMAX
          BEGINO
          BKDWN
          COUNT
          FINISH
          GETSYM
          IACON
          IHCON
          INDEX
          IOPRBD
          LABOUT
          NP2OUT
```

```
          NPUNCH
          NWRITE
          PACK
          PHASE6
          Q8PRMS
          RBDX
          RBPK
          SYMSCN
          TABDEC
          UNPUNC
          ENDLOC
*T
*K,I8
*N,FORTD1,,,B
*K,I6,P8
*P
          FTN
          GOE
          AMOUT
          ADMAX
          BEGINO
          BKDWN
          CONV
          COUNT
          FINISH
          GETSYM
          IACON
          IHCON
          INDEX
          IOPRBD
          LABOUT
          NP2OUT
          NPUNCH
          NWRITE
          PACK
          PHASE6
          Q8PRMS
          RBDX
          RBPK
          SETPRT
          SYMSCN
          TABDEC
          UNPUNC
          ENDLOC
*T
*K,I8
*N,FORTE1,,,B
*K,I6,P8
```

```
*L,FTN
          FTN
*L,Q8IFRM
          Q8IFRM
*L,Q8FS
          Q8FS
*L,Q8TRAN
          Q8TRAN
*L,FLOT
          FLOAT
*L,Q8QINI
          Q8QINI
*L,Q8QEND
          Q8QEND
*L,Q8CMP1
          Q8CMP
*L,Q8RWBU
          Q8RWBU
*L,Q8ERRM
          Q8ERRM
*L,Q8DFNF
          Q8DFIO
*L,Q8QX
          Q8QX
*L,Q8QUNI
          Q8QUNI
*L,Q8FGET
          Q8FGET
*L,Q8MAGT
          Q8MAGT
*L,Q8QBCK
          TAPCON
*L,IOCK
          IOCK
*L,Q8PSE
          PSSTOP
*L,Q8PAND
          Q8PAND
*L,Q8EXP9
          Q8EXP9
*L,Q8EXP1
          Q8EXP1
*L,Q8AB
          Q8AB
*L,SIGN
          SIGN
*L,EXP
          EXPPRG
```

```
*L, SQRT
        SQRTF
*L, ALOG
        LNUPRG
*L, TANH
        TANH
*L, SIN
        SINCOS
*L, ATAN
        ARCTPG
*L, QSAVE
        Q8EXPN
*L, IFALT
        IFALT
*L, Q8FX
        FXFL
*L, Q8PREP
        Q8PRMS
*U
```

7.5.2 COSY SOURCE TAPE

The FORTRAN 2.0B source tape is in COSY, compressed, format. The programs are arranged in the following order:

1. Phase A programs written in FORTRAN
2. Phase B programs written in FORTRAN
3. Phase C programs written in FORTRAN
4. Phase D programs written in FORTRAN
5. Phase E programs written in FORTRAN
6. Phases A, B, C, D, and E programs in assembly language
7. Object library programs written in FORTRAN
8. Object library programs written in assembly language

To assemble or compile a program, convert from COSY format into hollerigh format and then work with the hollerith tape. Following is a list of the FORTRAN routine sequence numbers and COSY deck names.

Phase A Programs Written in FORTRAN

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A1	CNVT	WCNVT	A1, A2, A3, A4, A5, B1
A2	GPUT	WGPUT	A1, A2, A3, A4, A5
A3	†SYMBOL	WSMBL1	A1, A2, A3, A4, A5

† Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A4	GETF	WGETF1	A1, A2, A3, A4, A5
A5	GNST	WGNST	A1, A2, A3, A4, A5
A6	OUTENT	WOTENT	A1, A2, A3, A4, A5
A7	PHASEA	WPHSEA	A1, A2, A3, A4, A5
A8	PLABEL	WPLBEL	A1, A2, A3, A4, A5
A9	Q8QBDS	WQ8QBS	A1
A10	RDLABL	WRLABL	A1, A2, A3, A4, A5
A11	STCHAR	WSCHAR	A1, A2, A3, A4, A5
A12	TYPE	WTYPE	A1, A2, A3, A4, A5
A13	ARITH	WARITH	A4
A14	COMNPR	WCMNPR	A2
A15	DIMPR	WDIMPR	A2
A16	SUBSCR	WSBSCR	A4
A17	TYPEPR	WTYPPR	A2
A18	BYEQPR	WBYEQ	A2
A19	CHECKF	WCKF	A2, A3
A20	FGETC	WFGETC	A2, A3
A21	FORK	WFORK	A2, A3
A23	SUBPPR	WSUB	A2
A234	EXRLPR	WEXRL	A2
A245	PEQVS	WPEQVS	A2
A256	PRNTNM	WPRTNM	A2
A267	†PUNT	WPUNT1	A3
A278	†SYMSCN	WSYMS1	A2, B1
A289	ENDDO	WENDDO	A1, A2, A3, A4, A5
A30	CONSUB	WCONSB	A2, A3
A31	DATAPR	WDATA	A2, A3
A32	ASGNPR	WASGN	A3
A33	BDOPR	WBDOPR	A3, A5
A34	CFIVOC	WCFVOC	A1, A2, A3, A4, A5

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A345	CKIVC	WCKVC	A3, A5
A356	CKNAME	WCKNAM	A1, A2, A3, A4, A5
A367	IOSPR	WIOSPR	A5
A38	ERBPR	WERBPR	A3
A389	MODMXR	WMODX	A3
A40	ASEMPR	WASEMP	A3
A41	TREE	WTREE	A4
A42	ARAYSZ	WARAY	A3, B1
A43	CPOLLP	WLOOP	A3
Phase B Programs Written in FORTRAN			
A50	DUMMY	WDUMMY	B1
A51	FCMSTK	WFCMK	B1
A52	KCPART	WKCPRT	B1
A53	KOUTPT	WKOTPT	B1
A54	KPCSTK	WKPCK	B1
A55	KPC3PR	WKPC3	B1
A56	KSYMGN	WKSYM	B1
A57	LABKPC	WLBKPC	B1
A58	LABLER	WLABLR	B1
A59	†PUNT	WPUNT2	B1
A60	†SYMBOL	WSMBL2	B1
A61	TSALOC	WTSLOC	B1
A62	ASSEM	WASSEM	B1
A63	BANANA	WBANAN	B1
A64	BGINDO	WBGNDO	B1
A65	END	WEND	B1
A66	ENTCOD	WENTCD	B1
A67	HELEN	WHELEN	B1
A68	INXRST	WXRST	B1
A69	NOPROC	WNOPR	B1
A70	PHASEB	WPHSEB	B1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A71	READIR	WRDIR	B1
A72	SUBFUN	WSUBFN	B1
A73	ACP	WACP	B1
A74	AFIDL	WAFIDL	B1
A75	ASUPER	WASPER	B1
A76	CGOTO	WCGOTO	B1
A77	FINK	WFINK	B1
A78	INTRAM	WTRAM	B1
A79	PARTSB	WPRTSB	B1
A80	SUBPR1	WSUB1	B1
A81	SUBPR2	WSUB2	B1
A82	SUBPR3	WSUB3	B1
A83	ARITHR	WRITHR	B1
Phase C, D, and E Programs Written in FORTRAN			
B1	†BKDWN	WBKDN1	C1
B2	BLDUP	WBLDUP	C1
B3	BSS	WBSS	C1
B4	CHKWD	WCHKWD	C1
B5	CHOP	WCHOP	C1
B6	CL12	WCL12	C1
B7	CON	WCON	C1
B8	†COUNT	WCNT1	C1
B9	DATAST	WDATST	C1
B10	†GETSYM	WGSYM1	C1, D1, E1
B11	INOUT	WINOUT	C1
B12	IXOPT	WIXOPT	C1
B13	PHASEC	WPHSEC	C1
B14	LABEL	WLABEL	C1
B15	LABIN	WLABIN	C1
B16	QXLD	WQXLD	C1
B17	REED	WREED	C1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
B18	†SKIP	WSKIP	C1
B19	†SYMSCN	WSYMS2	C1
B20	†AMOUT	WAMOT1	D1
B21	†ADMAX	WMAX1	D1
B22	†BEGINO	WGINO1	D1
B23	†BKDWN	WBKDN2	D1
B24	†COUNT	WCNT2	D1
B25	†FINISH	WFIN1	D1
B26	†IACON	WIACN1	D1
B27	†IHCON	WIHCN1	D1
B28	†INDEX	WDEX1	D1
B29	†LABOUT	WLABT1	D1
B30	†NP2OUT	WNP2T1	D1
B31	†NPUNCH	WNPUN1	D1
B32	†NWRITE	WNRIT1	D1
B33	†PHASE6	WPHS61	D1
B34	†RBDX	WRBDX1	D1
B35	†RBPK	WRBPK1	D1
B36	†SYMSCN	WSYMS3	D1, E1
B37	†TABDEC	WDEC1	D1
B38	†UNPUNC	WUNPC1	D1
B39	†AMOUT	WAMOT2	E1
B40	†ADMAX	WMAX2	E1
B41	†BEGINO	WGINO2	E1
B42	†BKDWN	WBKDN2	E1
B43	†COUNT	WCNT3	E1
B44	†FINISH	WFIN2	E1
B45	†IACON	WIACN2	E1
B46	†IHCON	WIHCN2	E1
B47	†INDEX	WDEX2	E1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
B48	†LABOUT	WLABT2	E1
B49	†NP2OUT	WNP2T2	E1
B50	†NPUNCH	WNPUN2	E1
B51	†NWRITE	WNRIT2	E1
B52	†PHASE6	WHPS62	E1
B53	†RBDX	WRBDX2	E1
B54	†RBPK	WRBPK2	E1
B55	†SETPRT	WSPRT	E1
B56	†TABDEC	WDEC2	E1
B57	†UNPUNC	WUNPC2	E1
Compiler Programs Written in FORTRAN			
B60	FTN	WFTNB	A1, A2, A3, A4, A5, B1, C1, D1, E1
B61	GOA	WGOA	A1, A2, A3, A4, A5
B62	†CONV	WCONV1	A1, A2, A3, A4, A5
B63	DIAG	WDIAG	A1, A2, A3, A4, A5
B64	EXP9	WEXP9	A1, A2, A3, A4, A5
B65	FLOAT	WFLOAT	A1, A2, A3, A4, A5
B66	GETSYM	WGSYM2	A1, A2, A3, A4, A5, B1
B67	IOPRBA	WIOPRA	A1, A2, A3, A4, A5
B68	PACK	WPACK	A1, A2, A3, A4, A5, D1, E1
B69	Q8PRMS	WQ8P	A1, A2, A3, A4, A5, B1, C1, D1, E1
B70	STORE	WSTORE	A1, A2, A3, A4, A5, B1
B71	LOCLA1	WLA1	A1
B72	DUMYA1	WDA1	A1
B73	GETC	WGETC1	A1, A2, A3, A4, A5
B74	IGETCF	WIGTCF	A1, A2, A3, A4, A5
B75	OPTION	WOPT	A1, A2, A3, A4, A5
B76	ENDLOC	WELOC	A1, A2, A3, A4, A5, B1, C1, D1, E1
B77	LOCLA2	WLA2	A2
B78	DUMYA2	WDA2	A2
B79	LOCLA3	WLA3	A3

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
B80	DUMYA3	WDA3	A3
B81	LOCLA4	WLA4	A4
B82	DUMYA4	WDA4	A4
B83	LOCLA5	WLA5	A5
B84	DUMYA5	WDA5	A5
B85	GOB	WGOB	B1
B86	IOPRBB	WIOPRB	B1
B87	GOC	WGOC	C1
B88	IOPRBC	WIOPRC	C1
B89	GOOD	WGOOD	D1
B90	IOPROBD	WIOPRD	D1, E1
B91	GOE	WGOE	E1
B92	†CONV	WCONV2	E1

Object Library Programs Written in FORTRAN

Deck Identifier	Program	Deck Name
Q01	Q8IFRM	Q8IFRM
Q02	Q8FS	Q8FS
Q03	Q8TRAN	Q8TRAN

Object Library Programs Written in Assembly Language

Deck Identifier	Program	Deck Name
Q04	†FLOAT	FLOAT2
Q05	Q8QINI	Q8QINI
Q06	Q8QEND	Q8QEND
Q07	Q8CMP	Q8CMP
Q08	Q8RWBU	Q8RWBU

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

Deck Identifier	Program	Deck Name
Q09	Q8ERRM	Q8ERRM
Q10	Q8DFIO	Q8DFIO
Q11	Q8QX	Q8QX
Q12	Q8QUNI	Q8QUNI
Q13	Q8FGET	Q8FGET
Q14	Q8MAGT	Q8MAGT
Q15	TAPCON	TAPCON
Q16	IOCK	IOCK
Q17	PSSTOP	PSSTOP
Q18	Q8PAND	Q8PAND
Q19	Q8EXP9	Q8EXP9
Q20	Q8EXP1	Q8EXP1
Q21	Q8AB	Q8AB
Q22	SIGN	SIGN
Q23	EXPPRG	EXPPRG
Q24	SQRTF	SQRTF
Q25	LNUPRG	LNUPRG
Q26	TANH	TANH
Q27	SINCOS	SINCOS
Q28	ARCTPG	ARCTPG
Q29	Q8EXPN	Q8EXPN
Q30	IFALT	IFALT
Q31	FXFL	FXFL

7.5.3 COMPILER PROGRAM ORDER

The compiler consists of four passes over the source code or its equivalent, accomplished in four phases called A, B, C, and D/E. (The fourth pass is performed by either Phase D or Phase E, depending upon whether the user wants to use an assembly language listing output.) Each phase consists of a root which is core-resident throughout the phase, and zero or more local subroutine groups which share the same core area and are read from disc as needed.

Phase A reads the source input, converts it to statements expressed in an internal code, and assigns a statement number to the statement.

Phase B reads the output of Phase A and generates pseudo code from it. (Pseudo code is similar to assembler input except that the index to be used in an indexed instruction and the addressing mode are not specified.)

Phase C and D/E are a two-pass assembler. The output from Phase B is read. Index registers are optimally assigned. One word relative addressing is maximized. Relocatable binary output and an assembly listing are produced.

Pass	Root	Local
FORTA1	A	1
FORTA2	A	2
FORTA3	A	3
FORTA4	A	4
FORTA5	A	5
FORTB1	B	
FORTC1	C	
FORTD1	D	
FORTE1	E	

Pass A

Root:

FTN
GOA
CFIVOC
CKNAME
CNVT
CONV
DIAG
EXP9
FLOAT
GETC
GETF
GETSYM
GPUT
IGETCF
IOPRBA
PACK
Q8PRMS
RDLABL
STORE
SYMBOL
ENDDO
GNST

```
OPTION
OUTENT
PHASEA
PLABEL
STCHAR
TYPE

Local 1:

LOCLA1
DUMYA1
Q8QBDS
ENDLOC

Local 2:

LOCLA2
DUMYA2
BYEQPR
CHECKF
COMNPR
CONSUB
DATAPR
DIMPR
EXRLPR
FGETC
FORK
PEQVS
PRNTNM
SUBPPR
SYMSCN
TYPEPR
ENDLOC

Local 3:

LOCLA3
DUMYA3
ARAYSZ
ASEMPR
ASGNPR
BDOPR
CHECKF
CKIVC
CONSUB
CPLOOP
DATAPR
FGETC
FORK
ERBPR
```

MODMXR
PUNT
ENDLOC

Local 4:

LOCLA4
DUMYA4
ARITH
SUBSCR
TREE
ENDLOC

Local 5:

LOCLA5
DUMYA5
BDOPR
CKIVC
IOSPR
ENDLOC

Pass B

Root:

FTN
GOB
CNVT
DUMMY
FCMSTK
GETSYM
IOPRBB
KCPART
KOUTPT
KPCSTK
KPC3PR
KSYMGN
LABKPC
LABLER
PUNT
Q8PRMS
STORE
SYMBOL
TSALOC
ARAYSZ
ASSEM
BANANA
BGINDO
END

ENTCOD
HELEN
INXRST
NOPROC
PHASEB
READIR
SUBFUN
SYMSCN
ACP
AFIDL
ASUPER
CGOTO
FINK
INTRAM
PARTSB
SUBPR1
SUBPR2
SUBPR3
ARITHR
ENDLOC

Pass C

Root:

FTN
GOC
BKDWN
BLDUP
BSS
CHKWD
CHOP
CL12
CON
COUNT
DATAST
GETSYM
INOUT
IOPRBC
IXOPT
LABEL
LABIN
PHASEC
Q8PRMS
QXLD
REED
SKIP
SYMSCN
ENDLOC

Pass D

Root:

FTN
GOOD
AMOUT
ADMAX
BEGINO
BKDWN
COUNT
FINISH
GETSYM
IACON
IHCON
INDEX
IOPRBD
LABOUT
NP2OUT
NPUNCH
NWRITE
PACK
PHASE6
Q8PRMS
RBDX
RBPK
SYMSCN
TABDEC
UNPUNC
ENDLOC

Pass E

Root:

FTN
GOE
AMOUT
ADMAX
BEGINO
BKDWN
CONV
COUNT
FINISH
GETSYM
IACON
IHCON
INDEX
IOPRBD
LABOUT

NP2OUT
NPUNCH
NWRITE
PACK
PHASE6
Q8PRMS
RBDX
RBPK
SETPRT
SYMSCN
TABDEC
UNPUNC
ENDLOC

7.5.4 COMPILER PROGRAM LENGTHS, COMMON LENGTHS, AND EXTERNALS

Program Name	Labeled Common	Blank Common	Program Length	Externals
CNVT	1641	3	62	
GPUT	1641	1236	41	
SYMBOL	1641	3	162	GETSYM WRITE SKIPIT
GETF	1641	1236	760	GETC GPUT DIAG EXP9 CNVT SYMBOL
GNST	1641	1236	446	CONV PACK WRITE IGETCF READ STCHAR EXIT DIAG SKIPIT
OUTENT	1641	1236	52	Q8PKUP Q8PREP WRITE
PHASEA	1641	1236	1259	WRITE GNST PLABEL TYPE

Program Name	Labeled Common	Blank Common	Program Length	Externals
				DIAG OUTENT PEQVS DIMPR COMNPR TYPEPR SUBPPR GETF STORE BYEQPR EXRLPR GETC DATAPR CHECKF ARITH GETSYM SYMBOL ASGNPR RDLABL ENDDO CPLOOP SKIPIT ERBPR IOSPR BDOPR STCHAR ASEMPR
PLABEL	1641	1236	86	RDLABL STORE DIAG
Q8QBDS	1641	0	0	
RDLABL	1641	1236	129	GETC DIAG CNVT SYMBOL
STCHAR	0	0	50	Q8PKUP Q8PREP
TYPE	1641	1236	510	Q8PKUP Q8PREP GETC GETF
ARITH	1641	1236	1669	LOCAL GETF

Program Name	Labeled Common	Blank Common	Program Length	Externals
				DIAG PUNT TREE GETSYM SUBSCR STORE GETC SYMBOL
COMNPR	1641	1236	150	GETF DIMPR DIAG
DIMPR	1641	1236	391	GETF DIAG STORE GETSYM
SUBSCR	1641	1236	701	GETF DIAG STORE PUNT GETSYM SYMBOL
TYPEPR	1641	1236	23	DIMPR
BYEQPR	1641	1236	499	GETF DIAG STORE GETSYM
CHECKF	1641	1236	160	FORK DIAG FGETC
FGETC	1641	1236	31	Q8PKUP Q8PREP GETC STCHAR
FORK	0	0	370	Q8PKUP Q8PREP FGETC
SUBPPR	1641	1236	181	GETF DIAG STORE
EXRLPR	1641	1236	94	GETF DIAG STORE

Program Name	Labeled Common	Blank Common	Program Length	Externals
PEQVS	1641	1236	991	SYMSCN GETSYM PRNTNM DIAG
PRNTNM	1641	0	141	Q8PKUP Q8PREP GETSYM WRITE
PUNT	0	1236	56	DIAG READ IGETCF SKIPIT
SYMSCN	1553	0	28	GETSYM
ENDDO	1641	1236	261	GETSYM OUTENT DIAG
CONSUB	1641	1236	135	DIAG GETF
DATAPR	1641	1236	400	GETF DIAG STORE PUNT CONSUB GETSYM
ASGNPR	1641	1236	70	RDLABL DIAG SYMBOL STORE GETC CKNAME
BDOPR	1641	1236	314	RDLABL DIAG STORE CKNAME CKIVC GETC GETSYM
CFIVOC	1641	1236	94	GETF DIAG STORE
CKIVC	0	1236	16	CFIVOC DIAG

Program Name	Labeled Common	Blank Common	Program Length	Externals
CKNAME	0	1236	16	CFIVOC DIAG
IOSPR	1641	1236	1699	LOCAL GETF DIAG SYMBOL STORE CKIVC RDLABL GETC STCHAR ENDDO BDOPR OUTENT
ERBPR	1641	1236	83	CKIVC DIAG
MODMXR	1641	1236	1116	CNVT SYMBOL STORE PUNT GETSYM
ASEMPR	1641	1236	434	GETC GETF DIAG STORE GETSYM RDLABL
TREE	1641	1236	1280	PUNT DIAG GETSYM
ARAYSZ	1641	0	106	Q8PKUP Q8PREP
CPLOOP	1641	550	165	GETSYM ARAYSZ
DUMMY	1547	1158	271	Q8PKUP Q8PREP GETSYM KSYMGN WRITE SKIPIT
FCMSTK	1547	1158	137	Q8PKUP Q8PREP KOUTPT

Program Name	Labeled Common	Blank Common	Program Length	Externals
KCPART	0	0	49	Q8PKUP Q8PREP
KOUTPT	1547	1158	18	WRITE
KPCSTK	1547	1158	955	Q8PKUP Q8PREP KCPART GETSYM DUMMY KOUTPT FCMSTK
KPC3PR	0	0	24	Q8PKUP Q8PREP KPCSTK
KSYMGN	1547	1158	72	Q8PKUP Q8PREP CNVT SYMBOL STORE
LABKPC	1547	1158	20	Q8PKUP Q8PREP KPCSTK
LABLER	1649	0	30	Q8PKUP Q8PREP KSYMGN
PUNT	1649	0	22	WRITE SKIPIT
SYMBOL	1649	3	157	GETSYM WRITE SKIPIT
TSALOC	1547	1158	139	Q8PKUP Q8PREP PUNT KSYMGN
ASSEM	1547	1158	103	KPCSTK LABKPC
BANANA	1547	1158	195	KPC3PR GETSYM LABKPC
BGINDO	1547	1158	265	PUNT LABLER

Program Name	Labeled Common	Blank Common	Program Length	Externals
				GETSYM KPC3PR LABKPC
END	1547	1158	72	LABKPC INXRST KPC3PR ENTCOD GETSYM
ENTCOD	1547	1158	169	KPC3PR LABKPC
HELEN	1547	1158	343	LABKPC KPC3PR ARAYSZ GETSYM SYMSCN
INXRST	1547	1158	20	KPC3PR
NOPROC	1547	1158	49	LABLER KPC3PR KOUTPT WRITE LABKPC
PHASEB	1547	1158	1109	LABLER LABKPC KPC3PR HELEN KPCSTK READIR TSALOC SUBFUN NOPROC ARITHR GETSYM ASUPER INXRST ENTCOD CGOTO BGINDO BANANA AFIDL PUNT ASSEM END

Program Name	Labeled Common	Blank Common	Program Length	Externals
READIR	1547	1158	88	Q8PKUP Q8PREP PUNT READ KPC3PR LABKPC
SUBFUN	1547	1158	103	GETSYM LABLER
ACP	1547	1158	1097	Q8PKUP Q8PREP PUNT INTRAM GETSYM KPC3PR KPCSTK TSALOC PARTSB FINK
AFIDL	1547	1158	90	Q8PKUP Q8PREP ASUPER KPC3PR
ASUPER	1547	1158	182	Q8PKUP Q8PREP PUNT SUBPR1 GETSYM SUBPR2 ACP
CGOTO	1547	1158	91	LABLER ASUPER KPC3PR KPCSTK LABKPC
FINK	1547	1158	181	KPCSTK LABLER KPC3PR LABKPC
INTRAM	1547	1158	473	PUNT KPCSTK KPC3PR TSALOC

Program Name	Labeled Common	Blank Common	Program Length	Externals
				GETSYM LABKPC
PARTSB	1547	1158	174	Q8PKUP Q8PREP KPCSTK GETSYM KPC3PR SYMBOL STORE
SUBPR1	1547	1158	62	Q8PKUP Q8PREP SUBPR3 TSALOC INTRAM KPC3PR
SUBPR2	1547	1158	141	Q8PKUP Q8PREP SUBPR3 KPC3PR GETSYM KPCSTK LABLER
SUBPR3	1547	1158	71	Q8PKUP Q8PREP ACP PARTSB
ARITHR	1547	1158	447	GETSYM SUBPR1 KPCSTK ASUPER KPC3PR
BKDWN	2993	1148	95	
BLDUP	2993	1148	67	
BSS	2993	1148	30	BLDUP
CHKWD	2993	1148	382	GETSYM
CHOP	2993	1148	544	GETSYM
CL12	2993	1148	185	GETSYM CHOP BLDUP
CON	2993	1148	55	BLDUP

Program Name	Labeled Common	Blank Common	Program Length	Externals
COUNT	2993	1148	23	
DATAST	2993	1148	167	GETSYM BLDUP INOUT
GETSYM	2993	0	164	WRITE READ
INOUT	2993	1148	111	REED BKDWN WRITE
IXOPT	2993	1148	315	CHKWD BKDWN QXLD GETSYM
PHASEC	2993	1148	847	WRITE READ SYMSCN REED DATAST GETSYM CHOP LABEL BLDUP INOUT BSS COUNT BKDWN CHKWD LABIN IXOPT CON CL12 QXLD SKIP
LABEL	2993	1148	34	BLDUP INOUT
LABIN	2993	1148	102	REED LABEL
QXLD	2993	1148	144	Q8PKUP Q8PREP CHOP BLDUP INOUT COUNT

Program Name	Labeled Common	Blank Common	Program Length	Externals
REED	2993	1148	93	READ
SKIP	2993	1148	86	CHOP BLDUP INOUT COUNT
SYMSCN	2993	1148	28	GETSYM
AMOUT	2993	1148	1498	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC NPUNCH
ADMAX	2993	1148	513	INDEX GETSYM TABDEC
BEGINO	2993	1148	272	NWRITE IHCON GETSYM IACON NPUNCH
BKDWN	2993	1148	105	INDEX
COUNT	2993	1148	23	
FINISH	2993	1148	367	NWRITE IHCON SYMSCN IACON NPUNCH
IACON	2993	1148	88	
IHCON	2993	1148	45	Q8PKUP Q8PREP
INDEX	2993	1148	28	Q8PKUP Q8PREP
LABOUT	2993	1148	223	Q8PKUP Q8PREP GETSYM

Program Name	Labeled Common	Blank Common	Program Length	Externals
				UNPUNC RBPK
NP2OUT	2993	1148	47	RBPK COUNT WRITE
NPUNCH	2993	1148	319	WRITE RESET READ
NWRITE	2993	1148	59	PACK WRITE
PHASE6	2993	1148	156	BEGINO READ INDEX AMOUT FINISH NPUNCH
RBDX	2993	1148	60	
RBPK	2993	1148	42	RBDX UNPUNC
SYMSCN	2993	1148	28	GETSYM
TABDEC	2993	1148	132	SYMSCN
UNPUNC	2993	1148	22	RBDX NPUNCH
AMOUT	2993	2090	1513	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC SETPRT CONV NPUNCH
ADMAX	2993	2090	513	INDEX GETSYM TABDEC

Program Name	Labeled Common	Blank Common	Program Length	Externals
BEGINO	2993	2090	321	NWRITE IHCON GETSYM IACON SETPRT NPUNCH SYMSCN
BKDWN	2993	2090	105	INDEX
COUNT	2993	2090	23	
FINISH	2993	2090	367	NWRITE IHCON SYMSCN IACON NPUNCH
IACON	2993	2090	88	
IHCON	2993	2090	44	Q8PKUP Q8PREP
INDEX	2993	2090	28	Q8PKUP Q8PREP
LABOUT	2993	2090	274	Q8PKUP Q8PREP GETSYM UNPUNC RBPK NWRITE IACON IHCON
NP2OUT	2993	2090	56	RBPK SETPRT COUNT WRITE
NPUNCH	2993	2090	319	WRITE RESET READ
NWRITE	2993	2090	59	PACK WRITE
PHASE6	2993	2090	156	BEGINO READ INDEX AMOUT

Program Name	Labeled Common	Blank Common	Program Length	Externals
				FINISH NPUNCH
RBDX	2993	2090	61	
RBPK	2993	2090	42	RBDX UNPUNC
SETPRT	2993	2090	379	IHCON IACON CONV NWRITE
TABDEC	2993	2090	124	SYMSCN
UNPUNC	2993	2090	22	RBDX NPUNCH

7.5.5 OBJECT LIBRARY PROGRAM ENTRY POINTS AND EXTERNALS

	Program Name	Entry Points	Externals
1.	FTN	FTN	
2.	Q8QINI	Q8QINI	Q8ERRM
		Q8UNIT	Q8INTB
		Q8SKIP	Q8EREM
			Q8QTOM
			Q8CMPO
			Q8MAGT
			Q8QEND
			Q8IFRM
			Q8IGP
			Q8CMP1
			Q8QUN2
			Q8DFIN
3.	Q8QEND	Q8QEND	Q8CMP1
			Q8QUN1
			Q8QUN2
			Q8UNIT
			Q8DFAD
4.	Q8CMP	Q8CMP0	Q8EREM
		Q8CMP1	Q8BEGB
		Q8DFAD	Q8LOCB
		Q8QENS	Q8CLRB
			Q8RINT
			Q8EOTT

	Program Name	Entry Points	Externals
5.	Q8RWBU	Q8BINB Q8LOCB Q8RWBU Q8INTB Q8BEGB Q8CLRB Q8RINT Q8IBUF	Q8CMP1 Q8EREM
6.	Q8ERRM	Q8ERRM Q8FERM Q8EREM	Q8LOCF Q8LOCB
7.	Q8DFIO	Q8DFNF Q8DFIN	Q8ERRM Q8EREM Q8QINI Q8DFAD Q8QENS
8.	Q8QX	Q8QTOM Q8QTRM Q8QX Q8MOVE Q8QY	Q8IFRM Q8BINB Q8QUN1 Q8UNIT
9.	Q8QUNI	Q8QUN1 Q8QUN2 Q8QUN3	Q8EREM
10.	Q8FGET	Q8FGET Q8FPUT Q8IGP Q8LOCF	
11.	Q8MAGT	Q8MAGT Q8EOTT	Q8EREM Q8QUN2 Q8COMI Q8QWND
12.	TAPCON	Q8QBCK Q8QFLE Q8QWND EOF	Q8EREM Q8CMP0 Q8CMP1 Q8QUN1 Q8QUN2 Q8QUN3 Q8IBUF
13.	IOCK	IOCK	Q8QUN1 Q8QUN3
14.	PSSTOP	Q8PSE Q8PSEN	Q8PAND

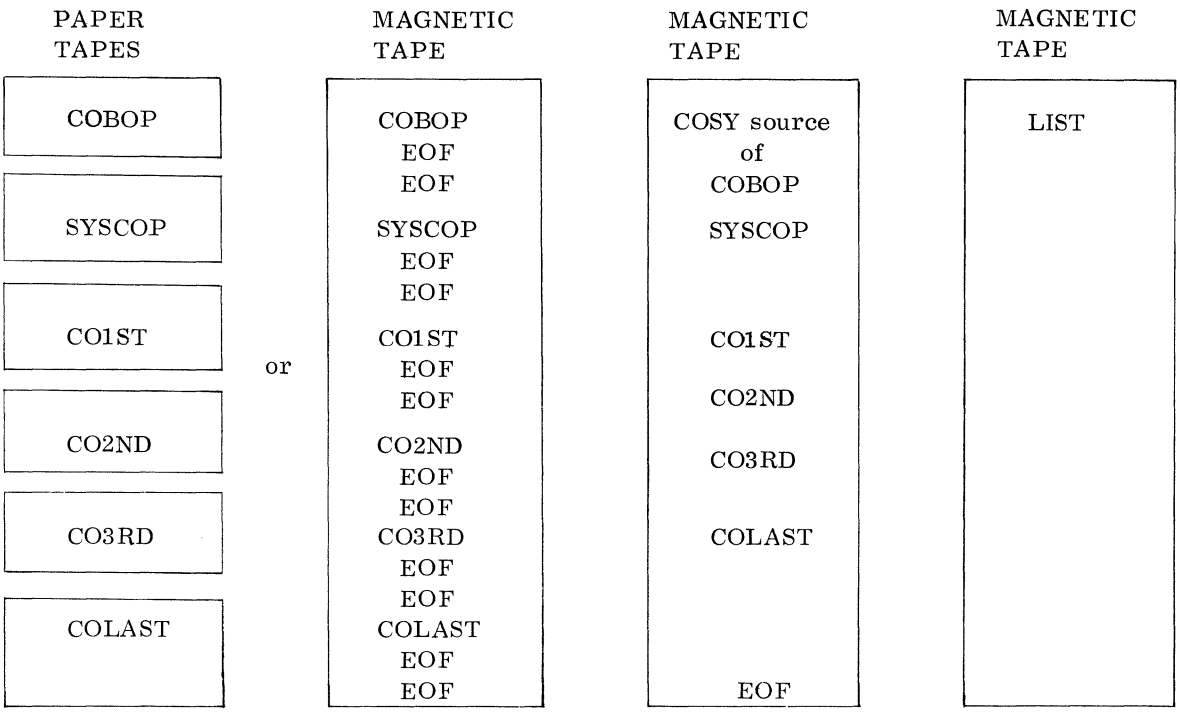
	Program Name	Entry Points	Externals
		Q8STP Q8STPN Q8COMI	
15.	Q8PAND	Q8PAND	
16.	Q8EXP9	Q8EXP9 Q8EXP1 Q8EXP2	FLOT
17.	Q8EXP1	Q8EXP1	FLOT Q8EXPT Q8EXP2
18.	Q8IFRM	Q8IFRM	Q8FS Q8TRAN Q8PKUP Q8PREP
19.	Q8FS	Q8FS	Q8SKIP Q8FGET Q8FERM Q8RWBU Q8FPUT Q8STP Q8PKUP Q8PREP
20.	Q8TRAN	Q8TRAN	Q8FS Q8RWBU Q8FERM Q8EXP9 Q8EXP1 Q8STP Q8PKUP Q8PREP Q8MOVE
21.	Q8AB	Q8AB ABS	FLOT
22.	SIGN	Q8SG SIGN	FLOT
23.	EXPPRG	EXP	FLOT
24.	SQRTF	SQRT	FLOT
25.	LNUPRG	ALOG	FLOT
26.	TANH	TANH	EXP FLOT

	Program Name	Entry Points	Externals
27.	SINCOS	SIN COS	FLOT
28.	ARCTPG	ATAN	FLOT
29.	Q8EXPN	Q8QF2I Q8QI2F Q8QF2F RETAD QSAVE	FLOT ALOG EXP
30.	FLOAT	FLOT	
31.	Q8PRMS	Q8PREP Q8PKUP	
32.	IFALT	IFALT	
33.	FXFL	Q8QFIX Q8FX Q8FLT Q8FLOT IFIX FLOAT	FLOT

7.6 SYSTEM CHECKOUT 1.0

Installation Tapes

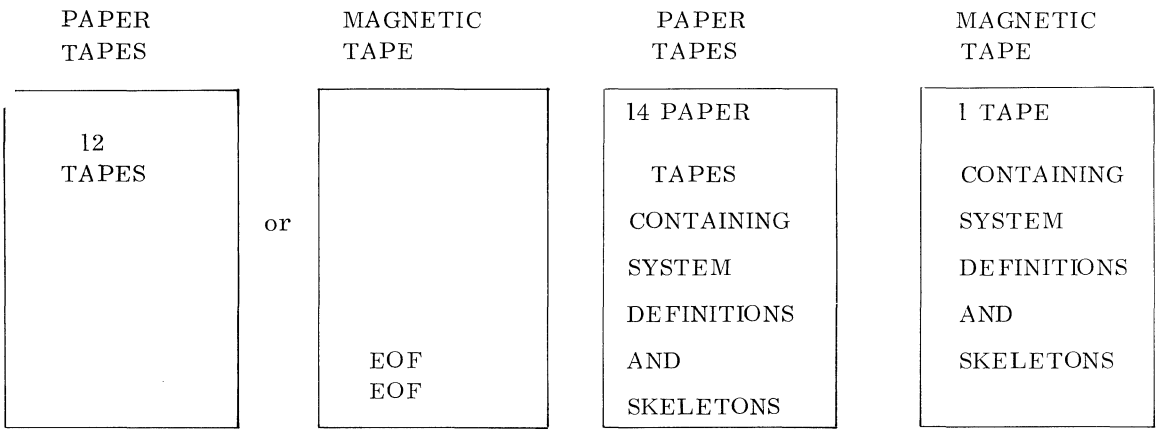
Optional Tapes



7.7 SYSTEM CONFIGURATOR

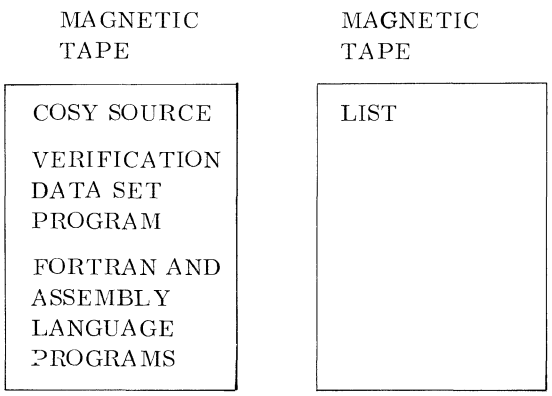
Installation Tapes

See section 7.7.1 for tape contents. System definitions and skeletons tape is further explained in section 7.7.2.



Optional Tapes

Section 7.7.3 contains further information on the COSY source tape.



7.7.1 RELEASE TAPE FORMAT

Installation Tapes

The System Configurator installation tape has the following format for magnetic tape. The paper tape format *K, I2, P8 replaces *K, I6, P8. Each deck name refers to a relocatable binary deck. Only the magnetic tape version is folllowed by two end-of-files.

```
*L, CONFIG
        CONFIG
*K, I6, P8
*P, F, GOCONF
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO1A
        OPTCHK
        INPREC
        MESSGS
        SCNOPT
        INITAL
        CONVRT
        CONTRL
        CORECT
        INSINP
        SCNREC
*T
*K, I8
*N, CONF1A, , , B
*K, I6
*P, F, GO1B
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO1B
        DEFINE
        PARCHK
        PAMCHK
        PARTIT
        SEARCH
```

```
        SCNREC
        INPREC
        CONTRL
        VALPRO
        VALCHK
        PICKUP
*T
*K,I8
*N,CONF1B,,,B
*K,I6
*P,F,GO1C
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO1C
        SYSDAT
        SYSINS
        INSINP
        INCINS
        GETCHR
        STOCHR
        WRTMMR
        RDSKEL
        INITCM
        COMMNT
*T
*K,I8
*N,CONF1C,,,B
*K,I6
*P,F,GO1D
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO1D
        SPECF1
        PARCHK
        BKCMVR
```

```
        SPCPAR
        SEARCH
        SCNREC
        CONTRL
        INPREC
        CORECK
        CORECT
        CONVRT
        INSINP
*T
*K,I8
*N,CONF1D,,,B
*K,I6
*P,F,GO1E
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO1E
        SPECF2
        PAMCH2
        INSURT
        INCINS
        INSINP
        SEARCH
        CORECT
        SCNREC
        CONTRL
        INPREC
        CNVTNO
        PICKUP
*T
*K,I8
*N,CONF1E,,,B
*K,I6
*P,F,GO1F
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
```

```
        GO1F
        VARPRO
        RNGCHK
        SCNREC
        OORECT
        VALCHK
        INPREC
        CONTRL
        CNVTNO
        PICKUP
*T
*K,I8
*N,CONF1F,,,B
*K,I6
*P,F,GO2
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO2
        PHASE2
        EQUIVA
        INSERT
        DELETE
        GETVAL
        CVTNUM
        GETNUM
        REDREC
        GETCHR
        GNSCHR
        STOCHR
        DECASC
        MMREAD
        OUTREC
        HICORE
        INTREG
        PICKUP
        P2NAM1
*T
*K,I8
*N,CONF2A,,,B
*K,I6
*P,F,GO2
        CONFIG
        GOCONF
        SCDKIO
        ERROR
```

```
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO2
        PHASE2
        EQUIVA
        DELETE
        GETVAL
        CVTNUM
        GETNUM
        REDREC
        GETCHR
        GNSCHR
        STOCHR
        DECASC
        MMREAD
        OUTREC
        MSKTBL
        FTNLVL
        SCHSTK
        PICKUP
        P2NAM2
*T
*K,I8
*N,CONF2B,,,B
*K,I6
*P,F,GO2
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO2
        PHASE2
        EQUIVA
        DELETE
        GETVAL
        CVTNUM
        GETNUM
        REDREC
        GETCHR
        GNSCHR
        STOCHR
```

```
        DECASC
        MMREAD
        OUTREC
        LUTBLS
        DGNTAB
        PICKUP
        P2NAM3
*T
*K,I8
*N,CONF2C,,,B
*K,I6
*P,F,GO2
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO2
        PHASE2
        EQUIVA
        INSERT
        DELETE
        GETVAL
        CVTNUM
        GETNUM
        REDREC
        GETCHR
        GNSCHR
        STOCHR
        DECASC
        MMREAD
        OUTREC
        OUTLNN
        FTNMSK
        PRESET
        PICKUP
        P2NAM4
*T
*K,I8
*N,CONF2D,,,B
*K,I6
*P,F,GO3A
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
```

```
        CALADR
        INCPTR
        GETFLE
        GO3A
        PHASE3
        PACKAG
        INSPGM
        DELPGM
        OUTORD
        XTCORE
        INPBIN
        OUTBIN
        UNLOAD
        GETVAL
        CVTNUM
        GETNUM
        GETCHR
        GNSCHR
        STOCHR
        BINASC
        PICKUP
        PAGEJT
        PRNTLN
        PACKLN
        NEWHDR
        STAPGM
        STAPCK
*T
*K,I8
*N,CONF3A,,,B
*K,I6
*P,F,GO3B
        CONFIG
        GOCONF
        SCDKIO
        ERROR
        DCTOAS
        GETITM
        CALADR
        INCPTR
        GETFLE
        GO3B
        INPBIN
        GETVAL
        CVTNUM
        GETNUM
        GETCHR
        GNSCHR
        BINASC
        PICKUP
        PAGEJT
```

```
        PRNTLN
        PACKLN
        OUTBIN
        STAEND
*T
*K,I8
*N, CONF3B, , , B
*U
```

7.7.2 SYSTEM DEFINITIONS AND SKELETONS TAPE

The system definitions and skeletons magnetic tape contains the system definitions, SYSDAT skeleton, and the system skeletons. It terminates with a double end-of-file. This is the tape released with MSOS 3.0 as referred to in section 7.1.1. The information on this tape also exists on eleven paper tapes and as a card deck for users installing on a paper tape system or a card system.

```
**          D E F I N I T I O N   L I S T
**
**
*S Y S T E M   H A R D W A R E   D E V I C E S
**
**      CORE MEMORY COMPONENT, INCLUDING THE INTERNAL INTERRUPT
*+CORE,
*       CORE STACKS                          (8)          (4,8)
*       PRIORITY LEVELS                      (16)         (8,16)
*       SCHEDULER ENTRIES                    (25)         (10,100)
*       EXTRA VOLATILE LOCATIONS             (0)          (0,100)
*       NON-SYSTEM CORE LOCATIONS            (0)          (0,$6000)
*       SWAP TIME IN SECONDS                 (0)          (0,600)
*       INTERNAL INTERRUPT LEVEL             (15)         (13,15)
*       CORE ALLOCATOR LEVEL                 (7)          (6,8)
*       ALLOCATABLE AREA LENGTHS (,,,,,,,,,,,)            (1,$6000)
*       PRESETS (,,,,,,,,,,,,,,,,,,,,,,,,,,,,)           (ANY)
**
**      1777 PAPER TAPE STATION
*+1777,
*       INTERRUPT LEVEL                      (10)         (8,12)
*       READER LOGICAL UNIT                  (2)          (2,127)
*       PUNCH LOGICAL UNIT                   (3)          (2,127)
*       READER ALTERNATE LOGICAL UNIT        ()           (2,127)
*       PUNCH ALTERNATE LOGICAL UNIT         ()           (2,127)
*       INTERRUPT LINE                       (1)          (1,15)
**
**      MASS MEMORY DRIVERS
*+MASS MEMORY,
*       COMMON PRIORITY                      (10)         (10,15)
*       NUMBER OF MASS MEMORY DRIVERS        (1)          (1,9)
*       DBLDRV REQUIRED                      (NO)         (YES,NO)
*       BUFFER SIZE                          ()           (1,$7FFF)
**
**      MASDRV REQUIRES A BUFFER THE SIZE OF THE LARGEST DRIVER
**      DBLDRV REQUIRES A BUFFER THE SIZE OF THE LARGEST TWO DRIVERS
**
**      IF ANY OF THE FOLLOWING DRIVERS ARE DESIRED TO BE MASS MEMORY
**      SET THE VALUE TO (YES)
**
**
**      IF THE 1713 IS SPECIFIED, IT MUST BE MASS MEMORY
*       1713 TELETYPE                        (NO)         (YES,NO)
*       1777 READER/PUNCH                    (NO)         (YES,NO)
*       1740/501                             (NO)         (YES,NO)
*       1723/430                             (NO)         (YES,NO)
*       1729-2                               (NO)         (YES,NO)
*       1726/405                             (NO)         (YES,NO)
*       MAGNETIC TAPE                        (NO)         (YES,NO)
**
```

```
**      ONLY ONE OF THE FOLLOWING MAGNETIC TAPE DRIVERS MAY BE CHOSEN
**
*       1731/601 BUFFERED                   (NO)          (YES,NO)
*       1731/601 UNBUFFERED                 (NO)          (YES,NO)
*       1732/608-9                          (NO)          (YES,NO)
**
**      1711/1712 TELETYPE
*+1711,
*       INTERRUPT LEVEL                     (10)          (8,12)
*       LOGICAL UNIT                        (4)           (2,127)
*       ALTERNATE LOGICAL UNIT              ()            (2,127)
**
**      1713 TELETYPE WITH READER/PUNCH
*+1713,
*       INTERRUPT LEVEL                     (10)          (8,12)
*       LOGICAL UNIT                        (4)           (2,127)
*       ALTERNATE LOGICAL UNIT              ()            (2,127)
*       READER LOGICAL UNIT                 (2)           (2,127)
*       READER ALTERNATE LOGICAL UNIT       ()            (2,127)
*       PUNCH LOGICAL UNIT                  (3)           (2,127)
*       PUNCH ALTERNATE LOGICAL UNIT        ()            (2,127)
**
**      1731 MAG TAPE CONTROLLER WITH 601 TAPE UNITS
*+1731/601,
*       INTERRUPT LINE                      (15)          (2,15)
*       INTERRUPT LEVEL                     (14)          (11,15)
*       TAPE UNITS                          (2)           (1,8)
*       LOGICAL UNITS                       (6,7,,,,,,)   (2,127)
*       ALTERNATE LOGICAL UNITS             (,,,,,,,)     (2,127)
*       EQUIPMENT NUMBER                    (7)           (0,15)
*       BUFFERED I/O                        (NO)          (YES,NO)
**
**      FOR THE 1731 BUFFERED DRIVER AREA 4 MUST BE SET TO
**      (MAXVAL)*3+1 OR $241 IF MAXVAL=192
**
**
**      1732 MAG TAPE CONTROLLER WITH 608 OR 609 TAPE UNITS
*+1732/608-9,
*       INTERRUPT LINE                      (3)           (2,15)
*       INTERRUPT LEVEL                     (11)          (9,13)
*       TAPE UNITS                          (1)           (1,8)
*       LOGICAL UNITS                       (6,7,,,,,,)   (2,127)
*       ALTERNATE LOGICAL UNITS             (,,,,,,,)     (2,127)
*       EQUIPMENT NUMBER                    (7)           (0,15)
*       BUFFERED I/O                        (NO)          (YES,NO)
*       608 TAPE UNITS                      (NO)          (YES,NO)
*************************************************************************
**      THE FOLLOWING IS USED TO GENERATE 608 AND 609 UNITS.
**      IF UNITS 6 AND 7 ARE SELECTED THE FOLLOWING WILL
**      GENERATE A 608 ON 6 AND A 609 ON 7.
**      MIXED 608/609                       (YES)
**      608 UNITS                           (YES,NO)
*************************************************************************
*       MIXED 608/609                       (NO)          (YES,NO)
*       608 UNITS                           (,,,,,,,)     (YES,NO)
```

```
**
**      THE 1732 DRIVER REQUIRES THAT AREA 4 BE SET TO
**      (P6)*4/3+2 OR $102 IF (P6)=192
**
**
**      1738 DISK CONTROLLER WITH 853 OR 854 DISK DRIVES
*+1738/853-4,
*       INTERRUPT LINE                  (4)         (2,15)
*       INTERRUPT LEVEL                 (9)         (8,10)
*       DISK DRIVES                     (1)         (1,2)
*       LOGICAL UNITS                   (8,)        (2,127)
*       ALTERNATE LOGICAL UNITS         (,)         (2,127)
*       EQUIPMENT NUMBER                (3)         (0,15)
*       853 DISK DRIVES                 (YES)       (YES,NO)
*************************************************************************
**      THE FOLLOWING WILL GENERATE ONE 853 AND ONE 854.
**      MIXED 853/854                   (YES)
**      853 UNITS                       (YES,NO)
*************************************************************************
*       MIXED 853/854                   (NO)        (YES,NO)
*       853 UNITS                       (,)         (YES,NO)
**
**      ONE NULL COMPONENT TO SAVE SPACE FOR POSSIBLE LATER USE
*+NULLOAC1
**
**      1751 DRUM
*+1751,
*       INTERRUPT LINE                  (6)         (2,15)
*       INTERRUPT LEVEL                 (12)        (10,14)
*       LOGICAL UNIT                    (12)        (2,127)
*       ALTERNATE LOGICAL UNIT          ()          (2,127)
*       MAXIMUM SECTORS                 ($AA9)      (1,$1554)
*       EQUIPMENT NUMBER                (2)         (0,15)
**
**      1723/430 CARD READER/PUNCH
*+1723/430,
*       COMMON INTERRUPT LINE           (10)        (2,15)
*       INTERRUPT LEVEL                 (14)        (12,15)
*       LOGICAL UNIT                    (10)        (2,127)
*       ALTERNATE LOGICAL UNIT          ()          (2,127)
*       EQUIPMENT NUMBER                (11)        (0,15)
**
**      1726/405 CARD READER
*+1726/405,
*       INTERRUPT LINE                  (5)         (2,15)
*       INTERRUPT LEVEL                 (8)         (8,15)
*       LOGICAL UNIT                    (5)         (2,127)
*       ALTERNATE LOGICAL UNIT          ()          (2,127)
*       EQUIPMENT NUMBER                (4)         (0,15)
*       BUFFERED I/O                    (NO)        (YES,NO)
**
**      BUFALC MODULE
**      THE FOLLOWING COMPONENT MUST BE SELECTED
**      IF BUFFERED I/O (1706) IS DESIRED
*+BUFALC,
**
```

```
**      1729-2 CARD READER
*+1729-2,
*       INTERRUPT LINE                      (11)        (2,15)
*       INTERRUPT LEVEL                     (13)        (11,15)
*       LOGICAL UNIT                        (11)        (2,127)
*       ALTERNATE LOGICAL UNIT              ()          (2,127)
*       EQUIPMENT NUMBER                    (12)        (0,15)
**
**      1740/501 LINE PRINTER
*+1740/501,
*       INTERRUPT LINE                      (12)        (2,15)
*       INTERRUPT LEVEL                     (10)        (8,12)
*       LOGICAL UNIT                        (9)         (2,127)
*       ALTERNATE LOGICAL UNIT              ()          (2,127)
*       FORTRAN LOGICAL UNIT                ()          (2,127)
*       ALTERNATE FORTRAN LOGICAL UNIT      ()          (2,127)
*       EQUIPMENT NUMBER                    (15)        (0,15)
**
**      1572 SAMPLE RATE GENERATOR
*+1572,
*       INTERRUPT LINE                      (2)         (2,15)
*       INTERRUPT LEVEL                     (13)        (11,15)
*       EQUIPMENT NUMBER                    (8)        (0,15)
*       CLOCK REQUENCY IN CYCLES/SECOND     (60)        (50,7680)
*       MAMIMUM SCHEDULES/TIMER PERIOD      (5)         (1,10)
*       OSCILLATOR FREQUENCY/CLOCK PERIOD (3333)        (1,4000)
**        THE ABOVE PARAMETER IS EQUAL TO OSCILLATOR FREQUENCY (IN CYCLES
**        PER SECOND) DIVIDED BY CLOCK FREQUENCY (IN CYCLES PER SECOND)
**
**      1573 LINE SYNC CLOCK
*+1573,
*       INTERRUPT LINE                      (2)         (2,15)
*       INTERRUPT LEVEL                     (13)        (11,15)
*       EQUIPMENT NUMBER                    (8)         (0,15)
*       CLOCK FREQUENCY IN CYCLES/SECOND    (60)        (50,7680)
*       MAXIMUM SCHEDULES/TIME PERIOD       (5)         (1,10)
**
**      DUMMY INPUT/OUTPUT DEVICE
*+DUMMY,
*       LOGICAL UNIT                        ()          (2,127)
*       DUMMY LEVEL                         (10)        (8,10)
**
**      STANDARD LOGICAL UNITS
*+STANDARD UNITS,
*       INPUT UNIT                          (2)         (2,127)
*       OUTPUT UNIT                         (3)         (2,127)
*       LIST UNIT                           (9)         (2,127)
*       COMMENT INPUT UNIT                  (4)         (2,127)
*       COMMENT OUTPUT UNIT                 (4)         (2,127)
*       LIBRARY UNIT                        (8)         (2,127)
*       SCRATCH UNIT                        (8)         (2,127)
**
**      INSERT COMPONENT FOR SYSTEM DATA PROGRAM
*+INSERT,
*       INPUT FROM LOGICAL UNIT             ()          (2,127)
*       REGION                              ()    (PROCES,MISCEL)
**
**
**
```

```
*C O R E   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
**
**      MSOS 3.0 MONITOR PACKAGE
*+MONITOR,
*       DISK WORD ADDRESSING                (NO)        (YES,NO)
**
**      INSERT COMPONENT FOR CORE RESIDENT FOREGROUND PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT             ()          (2,127)
*       ORDINAL NAME                        ()          (ANY)
**
**
**      FORTRAN RUNTIME REENTRANT LIBRARY
*+FTN RUNTIME LIBRARY,
*       ARITHMETIC FUNCTIONS                (YES)       (YES,NO)
*       MONITOR INTERFACE                   (YES)       (YES,NO)
*       FTN READ/WRITE STATEMENTS           (YES)       (YES,NO)
*       ENCODE/DECODE                       (YES)       (YES,NO)
*       REENTRANT FORTRAN LEVELS            (4,5,6,,,)  (2,15)
*       REENTRANT ENCODE/DECODE LEVELS      (4,5,6,,,)  (2,15)
**
*M A S S   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
**
**      JOB PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
*+JOB PROCESSOR,
*       MANUAL INTERRUPT PROCESSOR          (NO)        (YES,NO)
*       ON-LINE DEBUG PROGRAM               (NO)        (YES,NO)
**
**
**      INSERT COMPONENT FOR MASS RESIDENT FOREGROUND PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT             ()          (2,127)
*       ORDINAL NAME                        ()          (ANY)
```

```
*C O R E   R E S I D E N T   F O R E G R O U N D   P R O G R A M S
**
**      COMPONENT TO ANTICIPATE PROGRAMS THAT WILL BE ADDED IN THE FUTURE
*+PROGRAMS TO BE ADDED,
*       ESTIMATED NUMBER OF LOCATIONS     (0)          (0,$6000)
**
**      INSERT COMPONENT FOR CORE RESIDENT FOREGROUND PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT           ()           (2,127)
**
**
**
*P R O G R A M   L I B R A R Y   P R O G R A M S
**
*+FTN RUNTIME LIBRARY,
*       ARITHMETIC FUNCTIONS              (YES)        (YES,NO)
*       FORTRAN INPUT/OUTPUT              (YES)        (YES,NO)
*       MONITOR INTERFACE                 (YES)        (YES,NO)
*       ENCODE/DECODE                     (YES)        (YES,NO)
**
**      INSERT COMPONENT FOR PROGRAM LIBRARY PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT           ()           (2,127)
**
**
**
*TERMINATE
```

7.7.3 COSY SOURCE TAPE

The SYSCON COSY tape contains both FORTRAN and assembly language programs and terminates with an end-of-file mark.

The deck names for the FORTRAN programs are as follows:

VERIFY
BKCMVR
CALADR
CNVTNO
CONTRL
CONVRT
CORECK
CORECT
DCTOAS
DEFINE
GETCHR
GETITM
INCINS
INCPTR
INITAL
INITCM
INSURT
OPTCHK
PAMCHK
PAMCH2
PARCHK
PARTIT
RDSKEL
RNGCHK
SCNOPT
SCNREC
SEARCH
SPCPAR
SPECF1
SPECF2
STOCHR
SYSDAT
SYSINS
VALCHK
VALPRO
VARPRO
WRTMMR
PHASE2
CVTNUM
DECASC
DELETE
DGNTAB
EQUIVA
FTNLVL
FTNMSK

GETNUM
GETVAL
GNSCHR
HICORE
INSERT
INTREG
LUTBLS
MMREAD
MSKTBL
OUTLNN
PRESET
REDREC
SCHSTK
PHASE3
BINASC
DELPGM
INPBIN
INSPGM
NEWHDR
OUTORD
PACKAG
STAEND
STAPCK
STAPGM
XTCORE

The deck names for the assembly language programs are as follows:

COMMNT
CONFIG
ERROR
GETFLE
GOCONF
GO1A
GO1B
GO1C
GO1D
GO1E
GO1F
GO2
GO3A
GO3B
INPREC
INSINP
MESSGS
OUTBIN
OUTREC
PACKLN
PAGEJT
PICKUP
PRNTLN
P2NAM1

P2NAM2
P2NAM3
P2NAM4
SCDKIO
UNLOAD
SPACE

System Configurator Verification: The verification program is on the COSY source tape under the deck name VERIFY. Transfer the program to either paper tape, cards, or magnetic tape.

```
**              VERIFICATION  DECK  FOR  SYSCON
**              SPECIFICATION  LIST
**
*SYSTEM  HARDWARE  DEVICES
**
**   INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
*+1703,
**   1723/1724 PAPER TAPE PUNCH
*+1723,
**   1711/1712 TELETYPE
*+1711,
**   1738 DISK CONTROLLER WITH 853-4 DISK DRIVES
*+1738/853-4,
**
*CORE  RESIDENT  FOREGROUND  PROGRAMS
**
**   E006*2.1 MONITOR PACKAGE
*+MONITOR,
**
*MASS RESIDENT FOREGROUND PROGRAMS
**
**   JOB PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
*+JOB PROCESSOR,
**
**
*PROGRAM  LIBRARY  PROGRAMS
**
*+FTN RUNTIME LIBRARY,
**
*TERMINATE
```

INDEX

COMMENT SHEET

MANUAL TITLE 1700 MSOS 3.0 Installation Handbook

PUBLICATION NO. 60282700 REVISION

FROM: NAME:

BUSINESS ADDRESS:

COMMENTS:

This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number references and fill in publication revision level as shown by the last entry on the Record of Revision page at the front of the manual. Customer engineers are urged to use the TAR.

CUT ALONG LINE

AA3419 REV. 11/69 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Technical Publications Department
4201 North Lexington Avenue
Arden Hills, Minnesota 55112

ARH219

FOLD

FOLD

CONTROL DATA CORPORATION
Software Documentation
4201 North Lexington Avenue
St. Paul, Minnesota 55112

Pub. No. 60282700