GC33-5376-3
File No. S370-36

Systems

DOS/VS

System Control Statements

Release 32

IBM

Fourth Edition (November, 1975)

This is a major revision of, and obsoletes, GC33-5376-2. This edition applies to Version 5, Release 32, of the IBM Disk Operating System/Virtual Storage, DOS/VS, and to all subsequent versions and releases until otherwise indicated in new editions or Technical Newsletters.

This major revision includes information about the partition dump option, PARTDUMP, and the IBM 3344 and 3350 Direct Access Storage. Moreover, a number of maintenance changes have been included.

Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370 Bibliography, GC20-0001, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Laboratory, Publications Dept., P.O. Box 210, 703 Boeblingen, Germany. Comments become the property of IBM.

Preface

This manual is provided for those who need to know about the control statements that relate to the SCP (system control programming) of DOS/VS (Disk Operating System/Virtual Storage). The manual consists of the following:

- The sections Initial Program Loader and the Job Control describe these general programs of DOS/VS. These sections are of interest to anyone using the system, including system analysts, programmers, and operators. Detailed attention routine, job control statement, and job control command formats are given.
- The section POWER/VS is of interest to anyone who uses its spooling facilities to decrease the execution time of unit record I/O-bound jobs. It fully describes the control statements and commands for POWER/VS.

  The sections Linkage Editor and Librarian are of interest to persons responsible for maintaining the resident system. These sections fully describe the control statements for the linkage editor and librarian programs.
- The section System Buffer Load (SYSBUFLD) is of interest to DOS/VS users who have an IBM 3211, 3203, or 5203 Printers attached to their system. The section describes the purpose of SYSBUFLD and how to use it.
- Appendix A contains a summary of job control statements and commands.
- Appendix B contains a summary of the linkage editor.
- Appendix C contains a summary of the POWER/VS statements and commands.

Prerequisite publication:

DOS/VS System Management Guide, GC33-5371

Related publications are listed in the bibliography at the back of this book.

Contents

Figures

Introduction

This manual contains descriptions of control statements and commands. These statements and commands (except those involving TOLTEP) are descibed in detail in the following chapters.

INITIAL PROGRAM LOADER

Before a job can be entered into the system for execution, the supervisor must be read into the supervisor area of real storage and the job control program must be loaded into the virtual background partition. To do this, the operator starts the system by following the initial program load (IPL) procedure.

JOB CONTROL

After the system has been successfully started by means of the IPL procedure, it is ready to accept input for execution. Job control statements are entered via SYSRDR, job control commands via SYSLOG.

Job control runs in any virtual partition. It performs its functions between jobs and job steps. It is not present in the partition while a problem program is being executed.

The term END corresponds to the END key on the IBM 3210 and 3215 Console Printer-Keyboards (SYSLOG for IBM System/370 Models 135 and 145).

The term ENTER corresponds to the ENTER key on DOC (SYSLOG for IBM System/370 Model 115 and Model 125).

END and ENTER are used as message terminators on their respective systems. Thus, the term END/ENTER is used in this publication in reference to SYSLOG for the IBM System/370.

POWER/VS

POWER/VS is a reader/writer spooling system that provides automatic staging of unit-record input and output, class scheduling, and priority scheduling for all DOS/VS programs executed under its control.

POWER/VS resides in one partition of DOS/VS and controls from one to four other partitions, provided that these have a ower dispatching priority than that of the OWER/VS partition. POWER/VS is executed in virtual mode. Programs executed under control of POWER/VS may be executed not only in virtual mode but also in real mode.

In order to perform the required functions concurrently, POWER/VS is structured into a series of asynchronously executed tasks. All necessary task support is provided within POWER/VS and does not presuppose the presence of multitasking support within the DOS/VS supervisor.

TOLTEP

The Teleprocessing Online Test Executive Program (TOLTEP) can be considered as part of the IBM Virtual Telecommunications Access Method (VTAM). It is started when VTAM is started, and shares the VTAM partition as the highest priority subtask of VTAM. TOLTEP, VTAM, and the DOS/VS Online Tests form the Online Test System (OLTS). This system allows the testing of terminals supported by VTAM with minimum interference to other programs running in the system. TOLTEP for VTAM functions as an interface between the system and the test programs, and provides communication between the TOLTEP program and any user at a terminal recognized by VTAM and capable of alphameric character input and output. For a more detailed description of TOLTEP for VTAM and the operating procedures available, see the publication _DOS/VS and OS/VS TOLTEP for VTAM_, GC28-0663.

LINKAGE EDITOR

Prior to execution in storage, all programs must be placed in a core image library by the linkage editor.

The linkage editor prepares a program for execution by editing the output of a language translator into core image format. The linkage editor also combines separately assembled or compiled program sections or subprograms into phases.

LIBRARIAN

DOS/VS supports four types of libraries:

- Core image library - contains the output from the linkage editor (executable program phases).

- Relocatable library - contains the output of a language translator (object modules) which is used as input to the linkage editor.

- Source statement library - contains books (source language statements, macro definitions, and pre-edited macro definitions) used as input to a language translator.

- Procedure library - stores procedures, that is, sets of system control statements and (optionally) SYSIPT data. The procedures are used to build or complete job streams.

SYSTEM BUFFER LOAD (SYSBUFLD)

SYSBUFLD is a self-relocating special service control processing program for DOS/VS users with IBM 3211, 3203, and 5203 printers. It can be executed as a job or job step to load the Forms Control Buffer (FCB) and the Universal Character Set Buffer (UCB) of the 3211, 3203, and 5203.

CONTROL STATEMENT CONVENTIONS

The conventions used in this publication to illustrate control statements are as follows:

1. Uppercase letters and punctuation marks (except as described in items 3 through 5 below) represent information that must be coded exactly as shown.

2. Lowercase letters and terms represent information that must be supplied by the programmer.

3. Information contained within brackets [] represents an option that can be included or omitted, depending on the requirements of the program. Stacked options contained within brackets, for example

   $\begin{bmatrix}A\\B\end{bmatrix}$

   represent alternatives, one and only one of which may be chosen.

4. Options contained within braces {} represent alternatives, one of which must be chosen.

5. An ellipsis (...) indicates that a variable number of items may be included.

6. Underlined elements represent an assumed option in the event a parameter is omitted.

For this manual, alphameric characters are defined to include the following: A - Z, 0 - 9, @, $, and #.

In case of any difference between the conventions given in this manual for control program functions and those appearing in IBM-supplied DOS/VS component publications, observe the specific restrictions of the component.

SPECIAL DEVICES

The following devices that are mentioned are not available in the United States of America:

- IBM 1270 Optical Reader/Sorter

- IBM 1275 Optical Reader/Sorter.

Initial Program Loader

Operation of DOS/VS is initiated through an initial program load (IPL) procedure from the resident disk pack. The first record on track 0 is read into bytes 0-23 of real storage. The information read in consists of an IPL PSW and two CCWs, which in turn read and load the IPL.

The wait state is entered, at which time the user must indicate which supervisor he wants to be loaded.

Operating in the supervisor state, IPL reads the supervisor nucleus into low storage. If a read error is sensed while reading the supervisor nucleus, the wait state is entered and an error code is set in the first word of virtual storage. The IPL procedure must then be restarted.

After successfully reading in the supervisor nucleus, IPL performs these operations:

- Places the CPU in the EC mode.
- Sets the LUB table entry for SYSRES to point to the PUB entry of the channel and unit number of the resident drive.
- Places the processing unit in the wait state with all interruptions enabled.

  (The operator then causes an interrupt, which in turn will cause IPL to read its commands from the card reader, or from the printer-keyboard or DOC -- Display Operator Console.)
- Changes the PUB configuration, if indicated, by ADD or DEL commands or control cards.
- Scans the PUB table to determine whether or not one or more IBM 3211, 3203, and 5203 Printers are attached to the system.

The following describes the IPL commands: ADD, CAT, DEL, DPD, and SET.

ADD Command

To add a device to the PUB table, a command, read by the communication device (SYSLOG or SYSRDR), in the following format is required.

```
ADD X'cuu'[(k)],devicetype ┌,X'ss'     ┐
                           │,X'ssss'   │
                           └,X'ssssss' ┘
```

where:

X'cuu' = channel and unit numbers.

k can be specified as either S or a decimal number from 0 to 255:

S indicates that the device can be switched (that is, physically attached to two adjacent channels). The designated channel is the lower of the two channels.

0 to 255 indicates the priority of a device that cannot be switched, with 0 indicating the highest priority. If k is not given, the assumed priority is 255.

devicetype = actual device (2400T9, 1443, etc.). See type codes in Figure 1.

X'ss'
X'ssss'
X'ssssss' = device specifications (see ASSGN Statement). If absent, the following values are assigned:

X'C0' for 9-track tapes (2400, 3410 series)
X'D0' for 9-track tapes (3400 series)
X'90' for 7-track tapes
X'00' for nontapes
X'00', X'01', X'02', and X'03' are invalid as X'ss' for magnetic tape.

X'ss' specifies SADxxx (Set Address) requirements for IBM 2702 lines:

X'00' for SAD0
X'01' for SAD1
X'02' for SAD2
X'03' for SAD3

This information is not accepted on the ASSGN statements.

X'ss' is required for 1270, 1275, 1412, 1419, and 1419P device types. It specifies the external interrupt bit associated with magnetic ink or optical character readers. The setting X'01' through X'20' correspond to the external interrupt code in low real storage byte 87, bits 7 through 2 respectively. The corresponding external lines to which the control units are attached are as follows:

X'01'	byte 87 bit 7
X'02'	byte 87 bit 6
X'04'	byte 87 bit 5
X'08'	byte 87 bit 4
X'10'	byte 87 bit 3
X'20'	byte 87 bit 2

The X'ss' parameter specifies whether or not the error correction feature is present on an IBM 1018 Paper Tape Punch with 2826 Control Unit Model 1. X'ss' can be:

X'00'	no error correction feature
X'01'	error correction feature

For the 3704 and the 3705, X'ss' is required and must be specified as one of the following:

X'01'	Type 1 channel adapter
X'02'	Type 2 channel adapter.

For the 2703 of the Model 115 or 125, X'ss', X'ssss', or X'ssssss' is used to specify the line mode setting for a Start/Stop line or a BSC line.

The bit settings of the line mode specification for each line involved are explained in *IBM System/370 Model 115 Functional Characteristics*, GA33-1510, and in *IBM System/370 Model 125 Functional Characteristics*, GA33-1506, respectively. For POWER/VS RJE that uses a 2780 or 2770 terminal, you could, for instance, specify

X'008400'	for a non-switched line, or
X'00C400'	for a switched line

between the terminal and your Model 115 or 125.

The line mode setting is not accepted on the ASSGN statement. If a one or two byte value is specified, the specified value is right-justified and the rest of the three bytes is filled with zeros.

CAT Command

For VSAM files, the operator can issue the CAT command to assign the VSAM master catalog (logical unit SYSCAT) to a disk. The IPL assignment overrides a system generation SYSCAT assignment, if one was made, until the next IPL procedure. CAT may only be issued between the SET and DPD statements. The CAT command is entered from the card reader, the 3210, 3215, or the Display Operator Console. The format is:

```
CAT      UNIT=X'cuu'
```

where cuu is the channel and unit of the disk to be assigned to SYSCAT.

DEL Command

To delete a device from the PUB table, a command, read by the communication device (SYSLOG or SYSRDR), in the following format is required:

```
DEL      X'cuu'
```

where cuu is the channel and unit numbers of the device to be deleted.

DPD Command

The DPD command defines the page data set. The operation code of this command must always be specified during the IPL procedure. All operands are optional; they need be specified only if the required information was not supplied during system generation or if changes in the definition are desired. The DPD command must be the last command entered during the IPL procedure.

```
DPD   ┌TYPE={N}┐[,UNIT=X'cuu',CYL=xxx]
      └     {F}┘[,VOLID=xxxxxx]
```

The operands of the DPD command may be given in any order.

TYPE   TYPE=N indicates that the page data set need not be formatted and the extent limits have not been changed.

If TYPE=N is specified but the page data set does not exist or the extent limits have been changed, TYPE=N is ignored and the page data set is formatted during IPL. In this case, the UNIT and CYL operands must either have been supplied during system generation, or they must be specified in the DPD command.

TYPE=F indicates that the page data set is to be formatted during IPL. Formatting during IPL is required if the page data set is to be extended or if it is to be reallocated.

UNIT UNIT=X'cuu' specifies the channel and unit number (in hexadecimal) of the device that is to contain the page data set. If UNIT is specified, CYL must also be specified.

CYL CYL=xxx specifies the sequential number of the cylinder, relative to zero, where the page data set is to begin (in decimal). (The size of the page data set extent is calculated by the system.) If CYL is specified, UNIT must also be specified.

SYSVIS is the logical unit name of the page data set. It is created at IPL time from information provided by the system generation macro DPD and/or the IPL DPD command. The user specifies the beginning cylinder address and the system calculates the size of the disk extent which is a function of the VSIZE (virtual address area specified by the user) specification and the disk device type, as follows:

$\frac{VSIZE}{2}$ = number of pages (blocks of 2K bytes).

Disk Device Type	Blocks per Cylinder
2314	60
3330	114
3340	36

The allocation requires full cylinders.

The IBM-supplied supervisor does not contain any DPD macro parameters. Therefore, when you IPL, you must enter all operands of the DPD command, including TYPE=F. The IBM-supplied supervisor has a VSIZE of 128K and consequently one cylinder on a 3330 or two cylinders on a 2314 or 3340 are required.

VOLID VOLID=xxxxxx identifies the volume serial number (one to six alphabetic or numeric characters) of the disk pack that contains the page data set. If this operand is omitted both during system generation and in the DPD command, the volume serial number is not checked.

SET Command

Besides the DPD command, the only communication required at IPL time is the SET command. If any ADD or DEL commands are required, they must precede the SET command. The SET command must precede the DPD command. The SET command is entered via the communications device (3210, 3215, Display Operator Console or card reader) and is in the following format:

```
SET       [DATE=value1,CLOCK=value2]
          [,ZONE={EAST}/hh/mm]
          [      {WEST}      ]
```

value1 - Specifies the year, month, and day of the month in one of the following formats (depending on the installation's standard established during system generation):

mm/dd/yy
dd/mm/yy

value2 - Specifies the local time-of-day in the format hh/mm/ss.

EAST - Specifies that the installation is located at a geographical position east of Greenwich.

WEST - Specifies that the installation is located at a geographical position west of Greenwich.

hh/mm - A decimal value which indicates the difference in hours and minutes between local time and Greenwich Mean Time. hh may be in the range 0-12, mm in the range 0-59.

The parameters that have to be specified with the SET command depend upon the type of system and the type of communications device used. The following groups can be distinguished:

1. If the TOD clock is in the set state, the command may be given in one of the four possible forms:

```
SET
SET ZONE=
SET DATE=     ,CLOCK=
SET DATE=     ,CLOCK=     ,ZONE=
```

2. If the TOD clock is in the not-set state, the command must be given in either of two forms:

```
SET DATE=     ,CLOCK=
SET DATE=     ,CLOCK=     ,ZONE=
```

3. If the TOD clock is inoperative, the command must be given in the form:

```
SET DATE=   ,CLOCK=
```

Notes:

1. 3210/3215/DOC.

   If the TOD clock is in the set state, message 0I30I is printed. If the TOD clock is in the not-set state, message 0I31A is printed. If the TCD clock is inoperative, messages 0I32I and 0I31A are printed.

2. Card Reader.

   No messages are given.

3. CLOCK parameter.

   When using the CLOCK parameter of the SET command, the time-of-day security switch must be depressed to the ENABLE SET position.

4. Use of parameters.

   Although, when the TOD clock is in the set state, no parameters need to be specified with the SET command, it is strongly recommended to do so, for reasons of error recovery.

5. The date and time-of-day supplied in the SET command for systems that have TOD clock support should be realistic values, that is, the difference between local time and GMT may not be more than twelve hours, hh may not be more than 12, and mm may not be more than 59. The time-of-day clock should always contain the exact time (that is the time that has elapsed since January 1, 1900, 00.00 a.m.).

DEVICES

Figure 1 lists the device types available with DOS/VS. This figure shows the device type codes to be used in ADD commands when adding devices to the system.

Device Type Codes	Actual IBM Device	Device Type
2400T9	9-track Magnetic Tape Units (2400-series)	Magnetic Tape Devices
2400T7	7-track Magnetic Tape Units (2400-series)	
3410T9	9-track Magnetic Tape Units (3400-series)	
3410T7	9-track Magnetic Tape Units (3400-series)	
3420T9	9-track Magnetic Tape Units (3400-series)	
3420T7	7-track Magnetic Tape Units (3400-series)	
2495TC	2495 Tape Cartridge Reader	Tape Cartridge Reader
1442N1	1442N1 Card Read Punch	Card Read Punches
2520B1	2520B1 Card Read Punch	
2596	2596 Card Read Punch	
3525RP	3525 Card Read Punch (with optional read feature)	
2560	2560 Multifunction Card Machine	
5425	5425 Multifunction Card Unit	
2501	2501 Card Reader	Card Readers
2540R	2540 Card Reader	
3504	3504 Card Reader	
3505	3505 Card Reader	
2540P	2540Card Punch	Card Punches
2520B2	2520B2 Card Punch	
1442N2	1442N2 Card Punch	
2520B3	2520B3 Card Punch	
3525P	3525 Card Punch	
1403	1403 Printer	Printers
1403U	1403 Printer with UCS feature	
1443	1443 Printer	
2245	2245 Kanji Printer	
3211	3211 Printer	
5203	5203 Printer	
5203U	5203 Printer with UCS feature	
3203	3203 Printer	
3525P	3525 Card Punch (with optional print feature)	
1050A	3210, 3215 Console Printer-Keyboards	Printer-Keyboards
125D	Model 125 Integrated Display Operator Console	Display Units
125DP	Model 125 Integrated Display Operator Console with optional Console Printer	
UNSP	Unsupported Device	No burst mode on multiplexer channel
UNSPB	Unsupported Device	Burst mode on multiplexer channel
2311	2311 Disk Drive	DASD
2314	2314 Direct Access Storage Facility	
2314	2319 Disk Storage Facility	
3330	3330-1, 3330-2, or 3333-1 Disk Storage	
3330	3350 in 3330 mode	
3340R	3340 Disk Storage with Rotational Position Sensing (RPS) feature	
3340R	3344 Direct Access Storage	
3340	3340 Disk Storage without RPS feature	
2321	2321 Data Cell Drive	

Figure 1. Device Type Codes (Part 1 of 2)

Device Type Codes	Actual IBM Device	Device Type
3540	3540 Diskette Input/Output Unit	Diskette Input/Output Units
1419	1255 Magnetic Character Reader	MICR-Magnetic Ink Character Recognition Devices
1419	1259 Magnetic Character Reader	
1419	1419 Magnetic Character Reader	
1419P	1419 Dual Address Adapter Primary Control Unit	
1419S	1419 Dual Address Adapter Secondary Control Unit	
2701	2701 Data Adapter Unit	Teleprocessing Lines
2701	Integrated Communications Adapter (Model 135)	
2702	2702 Transmission Control Unit	
2703	2703 Transmission Control Unit	
2703	2703 Integrated Communications Adapter (Models 115 and 125)	
2703	3705 Communications Controller in Emulation Mode	
3704	3704 Communications Controller in Network Control Mode	
3705	3705 Communications Controller in Network Control Mode	
3791L	3791 Communications Controller	SNA Cluster Controller
2955	2955 Data Adapter Unit	Data Link for RETAIN
1017	1017 Paper Tape Reader with 2826 Control Unit Model 1	Paper Tape Readers
1017TP	1017 Paper Tape Reader with 2826 Control Unit Model 2	
2671	2671 Paper Tape Reader	
1018	1018 Paper Tape Punch with 2826 Control Unit Model 1	Paper Tape Punches
1018TP	1018 Paper Tape Punch with 2826 Control Unit Model 2	
1419	1270 Optical Reader Sorter	Optical Readers
1419P	1275 Optical Reader Sorter Primary Control Unit	
1419S	1275 Optical Reader Sorter Secondary Control Unit	
1287	1287 Optical Reader	
1288	1288 Optical Page Reader	
3881	3881 Optical Mark Reader	
3886	3886 Optical Character Reader	
2260	2260 Display Station or 1053 Printer	Display Station
3277	3277 Display Station or 3284-86 Printer	
3277B	3277 Display Station or 3284-86 Printer running in burst mode on the MPX channel	
7770	7770 Audio Response Unit	Audio Response Units

Figure 1. Device Type Codes (Part 2 of 2)

Job Control

This chapter contains descriptions, formats, and usages of the job control commands and statements, and attention routine commands, which are identified as follows:

job control statement - JCS

job control command - JCC

attention routines - AR

Figure 2 contains the commands and statements grouped by function, and also indicates the programs or routines for which they are valid. An alphabetical overview of all statements and commands is given in Appendix A.

Formatting Statements and Commands

JOB CONTROL STATEMENTS

Job control statements conform to the general rules for formatting DOS/VS control statements.

Name. Two slashes (//) identify the statement as a control statement. They must be in columns 1 and 2. At least one blank must immediately follow the second slash. Exception: The end-of-job statement contains /& in columns 1 and 2, the end-of-data-file statement contains /* in columns 1 and 2, the end-of-procedure statement contains /+ in columns 1 and 2, and the comment statement contains * in column 1 and blank in column 2.

Operation. Describes the operation to be performed. It can be up to eight characters long. At least one blank follows its last character.

Operand. May be blank or may contain one or more entries separated by commas. The last term must be followed by a blank, unless its last character is in column 71. Any blank within the operand fields, except for fields contained within apostrophes, is considered an end-of-operand indication. No further processing of that card occurs.

Type of Command or Statement	Operation	Valid for		
		JCS	AR	JCC
Job Identification	JOB	X		
	/&	X		
	/+	X		
File Definition	DLAB	X		
	DLBL	X		
	EXTENT	X		
	TLBL	X		
	TPLAB	X		
	VOL	X		
	XTENT	X		
	/*	X		
Pass Information to Operator	*	X		
Pass Information to Program	DATE	X		
	LBLTYP	X		
	OPTION	X		
	OVEND	X		X
	UPSI	X		
Job Stream Control	BATCH		X	
	CANCEL		X	X
	PAUSE	X	X	X
	PRTY		X	
	START		X	
	STOP			X
	TPBAL		X	
	UNBATCH			N1
Setting System Parameters	ALLOC		X	X
	ALLOCR		X	X
	SET			X
Operator Communications	ALTER		X	
	DSPLY		X	
	DUMP		X	
	END or ENTER key		X	X
	ENDSD		X	
	IGNORE		X	X
	LOG		X	X
	MSG		X	
	MODE		X	
	NEWVOL		X	
	NOLOG		X	X
	ZONE	X		
N1 Valid only in a foreground partition.				

Figure 2. JCS, JCC, and AR by Function (Part 1 of 2)

Type of Command or Statement	Operation	Valid for		
		JCS	AR	JCC
Control of I/O System	ASSGN	X		X
	CLOSE	X		X
	DVCDN			X
	DVCUP			X
	HCLD			X
	LFCB		X	
	LISTIO	X		X
	LUCB		X	
	MAP		X	X
	MTC	X		X
	RESET	X		X
	ROD			X
	UCS			X
Execution of Program	EXEC	X		X
	RSTRT	X		

Figure 2. JCS, JCC, and AR by Function (Part 2 of 2)

JOB CONTROL AND ATTENTION COMMANDS

Job control commands and attention commands contain the operation code, at least one blank and then the specified parameters. The parameters are separated by commas. The operation code usually begins in column 1 of the command, but this is not required.

- In contrast to job control statements (JCS), which are normally entered by the programmer, commands (job control and attention) are normally entered by the operator.
- Job control commands (JCC) are issued between jobs or job steps and are entered through SYSRDR or SYSLOG. (Job control statements, on the other hand, are usually coded as part of the input stream and are entered through SYSRDR.)
- Attention commands (AR) can be issued at any time by pressing the request key on SYSLOG. Some of these commands can be issued only in a multiprogramming environment.
- Some AR commands are used in conjunction with DOS/VS VTAM network operator commands. This usage is described in DOS/VS VTAM Network Operating Procedures, GC27-0025.

JOB CONTROL STATEMENTS SUMMARY

All job control statements are essentially free form. Information starts in column 1 and cannot extend past column 71. Continuation cards are not recognized by job control. For the exception to this rule, see the descriptions of the DLAB, DLBL, and TPLAB statements.

Job control normally reads from the device identified by the symbolic name SYSRDR. However, job control statements can also be entered through SYSLOG, or they may be retrieved from the procedure library, where they were previously cataloged. A brief description of the job control statements follows.

ASSGN	Used at execution time to assign a specific device address to the symbolic unit name used.
CLOSE	Closes either a system or a programmer logical unit assigned to tape, disk, or diskette.
DATE	Contains a date that is put in the communications region.
DLBL[1]	Contains file label information for DASD or diskette label checking and creation.
EXEC	Indicates the end of job control statements for a job step and that a cataloged procedure is to be retrieved before a job step is executed.
EXTENT[1]	Defines each area, or extent, of a DASD file or diskette volume.
JOB	Indicates the beginning of control information for a job.
LBLTYP	Defines the amount of storage to be reserved at link-edit time for processing tape and nonsequential DASD file labels in the partition.
LISTIO	Used to get a listing of I/O assignments. Ignored by job control if SYSLST is not assigned.
MTC	Controls operations on logical units to IBM 2400/3400 series magnetic tapes.
OPTION	Specifies one or more of the job control options.
OVEND	Indicates that no more overwrite statements will follow for the respective procedure.
PAUSE	Causes a pause immediately after processing this statement.
RESET	Resets I/O assignments to the standard assignments.
RSTRT	Restarts a checkpointed program.
TLBL[1]	Contains file label information for tape label checking and writing.
UPSI	(User Program Switch Indicators) Allows the user to set program switches that can be tested.
ZONE	Initializes the zone field in the communications region.
/*	Indicates the end of a data file or the end of a job step.
/&	Indicates the end of a job.
*	Job control comments.
/+	Indicates the end of a procedure.

Programming support continues for the following job control statements provided in previous versions of the system.

DLAB[1]	Contains file label information for DASD label checking and creation.
TPLAB[1]	Contains file label information for tape label checking and writing.
VOL[1]	Used when a set of label information for magnetic tape file or a DASD file is specified. It is not required with the current DLBL, EXTENT, or TLBL statements.
XTENT[1]	Defines each area, or extent, of a DASD file. It is used in conjunction with the VOL and DLAB statements.

[1] The combination of DLAB, VOL, and XTENT statements should not be used to supply the extent and label information for the IBM 3330, 3333, and 3340. To supply this information, use the DLBL and EXTENT statements.

The combination of VOL and TPLAB statements contains file label information for tape label checking and writing. These two statements can be replaced by the TLBL statement.

Any job control statement other than these is recognized as an error. A message is issued so that the programmer or operator can correct the statement in error. Some of the errors recognized are:

- Invalid symbolic unit name.
- No space reserved in LUB table for a symbolic unit.
- Invalid device type.
- Invalid length of field.
- Invalid character.
- Missing /& statement.
- A volume (VOL) statement does not precede a label (DLAB or TPLAB) statement.
- An EXTENT statement does not immediately follow its associated DASD label (DLBL) statement.

Whenever an invalid statement is indicated, the statement must be reissued to be effective. For example, if an OPTION LINK is encountered without a SYSLNK assignment, the OPTION statement must be reentered after assigning SYSLNK.

SEQUENCE OF JCS AND JCC

The job control statements for a specific job always begin with a JOB statement and end with a /& (end-of-job) statement. A specific job consists of one or more job steps. Each job step is initiated by an EXEC statement. Preceding the EXEC statement are any job control statements necessary to prepare for the execution of the specific job step. The only limitation on the sequence of statements preceding the EXEC statement is that DLBL statements must immediately precede the corresponding EXTENT statements. If the DLBL and EXTENT statements for a private core image library, SYSIN, SYSIPT, SYSRDR, SYSLST, and/or SYSPCH are in the input stream (if the information is not contained on the label cylinder), they must precede the ASSGN SYSCLB statement. If the DLBL and EXTENT statements for a tempory SYSLNK are in the job stream, they should precede the OPTION LINK or OPTION CATAL statement.

The LBLTYP statement is used at link-edit time and must precede the // EXEC LNKEDT statement; for self-relocating programs, however, it is submitted immediately preceding the // EXEC statement for the program.

INDIVIDUAL FORMATS (JCS, JCC, AR)

Detailed descriptions of the formats and functions of individual JCS, JCC, and AR statements and commands follow in alphabetic sequence. If the JCC and AR formats coincide, this is indicated by a combined heading, JCC and AR Format.

Job Control Statements and Commands

ALLOC

The ALLOC command (Allocate Virtual Address Area) permits the operator to allocate address space in the virtual address area (not already allocated to the shared virtual area) to foreground partitions. The number of bytes to be allocated for a foreground partition is specified in 2K (2048 bytes) increments.

Note: ALLOC cannot be specified for a single-partition system.

JCC and AR Format

```
ALLOC F1=nK[,F2=nK][,F3=nK][,F4=nK]
```

The order of operands is arbitrary. At least one operand must be specified. The maximum number of operands is one less than the number of partitions specified during system generation in the NPARTS parameter of the SUPVR macro.
The value n should be an even integer. n must not be smaller than 64 for an active foreground partition. Any specification smaller than 64 is flagged and must be corrected; any uneven specification greater than 64 is rounded up to the nearest even integer.
Note also that the size of the shared virtual area (SVA) must be taken into account. (Refer to the SET command for more details.) If VSAM is to be executed from the SVA, a GETVIS area of approximately 30K is required in the partition. If VSAM is to be executed in the partition, a GETVIS area of approximately 200K is required in that partition.

The following considerations apply to storage allocation among foreground and background programs that run in virtual mode:

1. Allocation of virtual partitions starts at the address immediately preceding the SVA, which is at the high end of the virtual address area. The areas are always contiguous, with no gaps between the allocated partitions. If the size of one or more partitions is changed, the boundaries of the remaining partition(s) are moved accordingly.

2. The size of the virtual background partition is not allocated or changed explicitly, but is changed implicitly by allocating or changing the size of one or more virtual foreground partitions. Storage neither allocated to the virtual foreground partitions nor to the shared virtual area automatically belongs to the virtual background. The virtual background must always have at least 64K bytes. The partition can be stopped by the STOP command, but not unbatched by the UNBATCH command.

3. The size of the virtual foreground partitions must always be allocated or changed explicitly. The size of an unspecified virtual foreground partition is not changed. When changing the size of one or more virtual foreground partitions, the starting and/or end addresses of the remaining partition(s) change accordingly. The ALLOC command must therefore be used with care with regard to programs which are linked to specific load addresses and which cannot be relocated.

4. To delete a virtual foreground partition from the system, you must issue an ALLOC command specifying a size of 0K for the respective partition. Only an inactive partition can be deleted. A foreground partition is inactive if it has been unbatched (by the UNBATCH Command) or if it has never been activated (by the BATCH or START command).

5. The maximum permissible size of the virtual background partition is the size of the virtual address area minus the size of the SVA specified for the system. The maximum size allowed for a virtual foreground partition is the size of the virtual address area minus 64K for the virtual background and minus 64K for the SVA.

6. No allocation takes place when the ALLOC command would move the start address of a virtual partition upwards and/or the end address downwards while a program is running in that partition.

   Exception: The end address of the partition in which job control is processing an ALLOC command may be moved downwards so that a partition of 64K remains.

If the rules listed are violated, corresponding error messages (1P0nD or 1S0nD) are issued, indicating that the ALLOC command was rejected and that a correct command has to be entered.

ALLOCR

The ALLOCR command (Allocate Real Storage) permits the operator to allocate the real address area among foreground and background programs, that is, to allocate real partitions. The number of bytes to be allocated is specified in 2K increments.

Note: ALLOCR cannot be specified for a single-partition system.

JCC and AR Format

```
ALLOCR BGR=nK[,F1R=nK][,F2R=nK]
       [,F3R=nK][,F4R=nK]
```

The order of operands is arbitrary; at least one operand must be specified. The maximum number of operands is equal to the number of partitions specified during system generation in the NPARTS parameter of the SUPVR macro.

n must be an even integer. It may also be zero.

The following considerations apply to storage allocation in the real address area:

1. Allocation starts at the end of the supervisor. The areas are always contiguous, with no gaps between the allocated partitions. The main page pool is at the high end of the real address area. If the size of one or more real partitions is changed, the boundaries of the remaining partition(s) are moved accordingly. For this reason the ALLOCR command must be handled with care with regard to programs that are linked to specific load addresses in real storage and which cannot be relocated.

   Note: The maximum size of the real address area which can be allocated by the ALLOCR command is the area specified by the RSIZE parameter of the VSTAB macro at system generation (minus the size of the supervisor) or the real storage available in the hardware system being used, (minus the size of the supervisor), whichever is smaller.

2. The size of the page pool is not allocated or changed explicitly but is changed implicitly by allocating or changing the size of one or more real partitions. Storage not allocated to real partitions and not occupied by the supervisor belongs automatically to the main page pool.

   The minimum size of the main page pool is:

   - 18K minus the size of the smallest real partition if the smallest real partition is 18K or less and PFIX=NO (plus 2K if AP=YES). If the SDL (system directory list) is active, however, the main page pool must be at least 4K.
   - 18K if PFIX=YES (plus 2K if AP=YES).
   - 18K if phases from the SVA are to be executed (plus 2K if AP=YES).

3. The size of each real partition must always be allocated or changed explicitly (including that of the real background partition). The size of an unspecified real partition is not changed.

   The size of each real partition may be changed arbitrarily; it may even be reduced to zero. Depending on the changes made, however, the boundaries of adjacent partitions change.

   The size of an active real partition can be reduced only via the ALLOCR job control command. When a real partition is inactive, either the ALLOCR job control command or the ALLOCR attention command may be used.

   The background partition is always active. A foreground partition is inactive if it has been unbatched (by the UNBATCH command) or if it has never been batched (by the BATCH or START command). Otherwise it is active. When a partition is increased, either the size of another partition or the main page pool is decreased. A partition can be decreased only if it is inactive.

4. A program running in a virtual partition may fix pages (via the PFIX assembler macro) only if the attached real partition in a multi-partition system has a size greater than zero. It may fix as many pages as the real partition contains page frames.

   In a single-partition system, a program running in virtual mode may fix all pages with the exception of the 18K required for the minimum main page pool,

5. Real storage is not allocated when an ALLOCR command would move the start address of a real partition upwards and/or the end address downwards while a program is running in that partition, or when the real partition belongs to a virtual partition in which a program running in virtual mode may fix pages.

   Exception: The size of a real partition in whose attached virtual partition job control is processing an ALLOCR command may be changed arbitrarily; it may even be reduced to zero.

If the rules listed are violated, corresponding error messages (1P0nD or 1S0nD) are issued, indicating that the ALLOCR command was rejected and that a correct command has to be entered.

ALTER

The ALTER command allows the operator to alter 1 to 16 bytes of virtual storage, starting at the specified hexadecimal address. After the command has been entered and the END/ENTER key pressed, the hexadecimal representation of the information to be placed in storage should be entered on the device assigned to SYSLOG. Two hexadecimal characters (0 through F) must be entered for each byte to be changed. If an odd number of characters is entered, the last character is ignored and its associated byte is unaltered.

AR Format

ALTER address

address The six-digit hexadecimal address, with leading zeros if necessary, to start storage alteration.

If the specified address is within the supervisor area or the shared virtual area (SVA), a message is issued and the operator has the option to cancel or to change the address.

If the specified address is within an invalid address range, the command is ignored and a corresponding informatory message issued.

If the 16 bytes to be altered cross the boundary from a valid to an invalid address space, only the bytes of the valid address space are altered and a corresponding informatory message is issued.

ASSGN

The ASSGN command or statement (Assign Logical Name) assigns a logical I/O unit to a physical device. Only DASD can be assigned to (shared by) more than one partition concurrently.

The general format for the ASSGN statement/command is as follows:

Name	Operation	Operand 1	Operand 2	Other Operands (Optional)
//	ASSGN	SYSxxx,	X'cuu'	,TEMP
			(address-list)	,PERM
(blank for JCC)			UA	,VOL=volserno
			IGN	,SHR
			SYSyyy	,X'ss'
			device-class	,ALT
			device-type	,H1
				,H2

The operands of the ASSGN card can be combined in various ways. The format is therefore broken down into groups:

- For any device
- For disks
- For diskettes
- For tapes
- For printers
- For card (read) punches
- For card readers.

The operands are processed in the sequence given for each combination.

For any device

```
[//] ASSGN SYSxxx,(X'cuu'        ) [,TEMP]
                  (address-list) ( [,PERM]
                  (SYSyyy        )
                  (UA            )
                  (IGN           )
```

For disks

```
[//] ASSGN SYSxxx,{X'cuu'          }[,TEMP][,VOL=volserno] [,SHR]
                  {(address-list)  }[,PERM]
                  {SYSyyy          }
                  {DISK            }
                  {2311            }
                  {2314            }
                  {3330            }
                  {3340            }
```

For diskettes

```
[//] ASSGN SYSxxx,{X'cuu'          }[,TEMP]
                  {(address-list)  }[,PERM]
                  {SYSyyy          }
                  {DISKETTE        }
                  {3540            }
```

For tapes

```
[//] ASSGN SYSxxx,{X'cuu'          }[,X'ss'][,TEMP][,VOL=volserno]
                  {(address-list)  }[,ALT  ][,PERM]
                  {SYSyyy          }
                  {TAPE            }
                  {2400T7          }
                  {2400T9          }
                  {3410T7          }
                  {3410T9          }
                  {3420T7          }
                  {3420T9          }
```

For printers

```
[//] ASSGN SYSxxx,{X'cuu'          }[,TEMP]
                  {(address-list)  }[,PERM]
                  {SYSyyy          }
                  {PRINTER         }
                  {1403            }
                  {1403U           }
                  {1443            }
                  {3203            }
                  {3211            }
                  {5203            }
                  {5203U           }
```

For card (read) punches

```
[//] ASSGN SYSxxx,{X'cuu'          }[,TEMP]
                  {(address-list)  }[,PERM]
                  {SYSyyy          }
                  {PUNCH           }
                  {1442N1          }
                  {1442N2          }
                  {2520B1          }
                  {2520B2          }
                  {2520B3          }
                  {2540P           }
                  {2560[,H1]       }
                  {    [,H2]       }
                  {2596            }
                  {3525P           }
                  {3525RP          }
                  {5425[,H1]       }
                  {    [,H2]       }
```

For card readers

```
[//] ASSGN SYSxxx, ⎧X'cuu'         ⎫ ⎡,TEMP⎤
                   ⎪(address-list) ⎪ ⎣,PERM⎦
                   ⎪SYSyyy         ⎪
                   ⎪READER         ⎪
                   ⎪1442N1         ⎪
                   ⎪2501           ⎪
                   ⎪2520B1         ⎪
                   ⎪2540R          ⎪
                   ⎨2560⎡,H1⎤      ⎬
                   ⎪    ⎣,H2⎦      ⎪
                   ⎪2596           ⎪
                   ⎪3504           ⎪
                   ⎪3505           ⎪
                   ⎪3525RP         ⎪
                   ⎪5425⎡,H1⎤      ⎪
                   ⎩    ⎣,H2⎦      ⎭
```

The job control statement (// ASSGN) is temporary. It remains in effect only until the next change in assignment or until the end of job, whichever occurs first. The job control command (ASSGN) is permanent. It remains in effect until the next permanent assignment, the DVCDN command, or re-IPL of the system, whichever occurs first. A CLOSE command to a system logical unit on disk or the 3540 diskette also removes a permanent assignment. See also the TEMP override of a permanent ASSGN and the PERM override of a temporary ASSGN.

At the completion of a job, a temporary assignment is automatically restored to the permanent assignment for the logical unit.

The entries in the operand field represent the following.

SYSxxx The symbolic unit name. It can be one of the following. (After supervisor generation, SYSCAT can only be assigned by the CAT command at IPL time.)

SYSRDR
SYSIPT[4]
SYSIN
SYSPCH
SYSLST
SYSOUT[1,2]
SYSLNK
SYSLOG
SYSSLB
SYSRLB
SYSREC
SYSCLB[1,3]
SYSnnn represents all the other symbolic units in the system. These units vary from SYS000 to SYSmax, where SYSmax represents the highest numbered programmer logical unit available for the system. SYSmax is equal to 255 minus (number of partitions multiplied by14). Each of these programmer logical units can be assigned to any partition without a prescribed sequence, except when using DAM. For a given partition, the maximum number of programmer logical units is equal to SYSmax minus the sum of all programmer logical units assigned to other partitions.

Notes:

1. Must be permanent assignments, for example, the TEMP option is not permitted for assigning these logical units.

2. Valid for a tape unit only.

3. Only if the system permits private core image libraries.

4. If SYSIPT is assigned to a tape unit, it should be a single file and a single volume.

If a system logical unit is assigned to a tape, DASD, or 3540 diskette, the unit should be closed (using the CLOSE command) before it can be reassigned.

When SYSOUT is assigned, the magnetic tape device must not be the permanent assignment of either SYSLST or SYSPCH. Before assigning a tape drive to a system output unit (SYSOUT, SYSLST, SYSPCH), all previous assignments of this tape drive to any system input units and to any programmer units (input or output) must be permanently unassigned. The assignment of SYSOUT must always be permanent. For detailed information on the use of SYSOUT, refer to chapter 5, section "Symbolic I/O Assignments" in the DOS/VS System Management Guide, GC33-5371. It is not possible to change the assignment of SYSLOG while a foreground partition is active.

If SYSLNK is assigned to a foreground partition(s), SYSCLB must also be assigned to the same partition(s). Whenever the DLBL and EXTENT information for SYSCLB changes, SYSCLB must be reassigned.

X'cuu' Indicates the channel and unit number (in hexadecimal).

c = 0 to 6 for the channel
uu= 00 to FE (0 to 254) in
    hexadecimal

(address-list)
You can specify a list of up to seven device addresses in the form X'cuu', separated by commas and enclosed in parentheses. In this case the system searches only the PUB entries referenced in the address list for a free unit, starting with the first specified device address. Once a free unit is found, it is assigned to SYSxxx for the job in which the assignment is made. For disks, if SHR is specified, the first unit in the list is assigned, even if previously assigned.

UA Indicates that the logical unit is to be unassigned. Any operation attempted on an unassigned device cancels the job.

IGN For certain American National Standard and DOS/VS COBOL problem programs (for sequential input files), and for FORTRAN the IGN option unassigns the specified logical unit, and ignores any subsequent logical IOCS command (OPEN, GET, etc.), issued for that unit. This allows you to disable a logical unit that is used in a program without removing the code for that unit. You can then execute the program as if the unit did not exist. This may be especially helpful when debugging a program.

For assembler language problem programs, IGN indicates that the logical unit is to be ignored. With files processed by logical IOCS, the OPEN to the file is ignored, the DTF table is not initialized (for example, IOREG, extent limits), and the IGNORE indicator is set on in the DTF table. It is your responsibility to check this indicator and bypass any I/O commands (GET, PUT, etc.) for this file.

The IGN option is not valid for SYSRDR, SYSIPT, SYSIN, and SYSCLB, nor for PL/I programs or QTAM teleprocessing devices. The ING option can be made temporary by specifying the TEMP option.

Additional information about ignore is in the OPEN (R) section of DOS/VS Supervisor and I/O Macros, GC33-5373. IGN restrictions for users of American National Standard and DOS/VS COBOL and of RPG II are given in the associated Program Product publications for the processor being used.

SYSyyy This may be any system or programmer logical unit as shown earlier under the description of SYSxxx. In this case, SYSxxx is assigned to the same device to which SYSyyy is assigned. This type of specification is particularly helpful in the case of disk, because the specification of SYSxxx, SYSyyy is considerably shorter than the full specification.

Example:

```
// ASSGN SYS001,2314,PERM,
         VOL=RAFT01,SHR
// ASSGN SYS003,SYS001
// ASSGN SYSLNK,SYS001
```

device-class
In this case the specification of READER, PRINTER, PUNCH, TAPE, DISK, or DISKETTE is allowed. Do not, however, use a generic assignment for a dummy device to be used as input or output devices in a POWER/VS supported partition. The system searches the PUB tables for the first unassigned unit of the specified device-class and assigns it to SYSxxx. This type of specification might be used if the exact configuration of the installation is not known or not important. For disks, if SHR is specified, the first unit of the specified device class is assigned, even if previously assigned. If your installation includes 3340 drives with and without the Fixed Head Feature, and you are using the 3348 Model 70F Data Module or the 3344 Direct Access Storage, you should not use device-class or device-type. Instead, use X'cuu' (or address list) to specify the drives with the feature, so as to avoid job cancellation. For more information, see the IBM 3340 Fixed Head Feature Users Guide, GA26-1632 or the Introduction to IBM 3350 Storage , GA26-1638.

The specific device types to which each device class applies are listed below.

READER	1442N1, 2501, 2520B1, 2540R, 2560, 2596, 3504, 3505, 3525RP, 5425
PRINTER	1403, 1403U, 1443, 3203, 3211, 5203, 5203U
PUNCH	1442N1, 1442N2, 2520B1, 2520B2, 2520B3, 2540P, 2560, 2596, 3525P, 3525RP, 5425
TAPE	2400T7, 2400T9, 3410T7, 3410T9, 3420T7, 3420T9
DISK	2311, 2314/2319, 3330/3333, 3340, 3340R
DISKETTE	3540

device-type
This can be the device code of any supported device-type, such as 2400T9 or 3525RP. In this case the PUB table of the specified device-type is searched for the first free unit. When a free unit is found, it is assigned to SYSxxx. Use this specification if you are interested only in the specific type of device, and not in the physical unit. For disks, if SHR is specified, the first unit of the specified device-type is assigned, even if previously assigned. If your installation includes 3340 drives with and without the Fixed Head Feature, and you are using the 3348 Model 70F Data Module or the 3344 Direct Access Storage, you should not use device-class or device-type. Instead, use X'cuu' (or address list) to specify the drives with the feature, so as to avoid job cancellation. For more information, see the IBM 3340 Fixed Head Feature Users Guide, GA26-1632 or the Introduction to IBM 3350 Storage, GA26-1638.

Figure 3 shows an example of how the PUB table is scanned with different types of tape specifications in the ASSGN statement/command.

PUB Table		Search Order for Specification		
Phys. Unit	Dev. Type	Tape	2400T9	X'380', X'381', X'183', X'284'
181	2400T9	1	1	
182	2400T9	2	2	
183	2400T9	3	3	3
281	2400T7	4		
282	2400T7	5		
283	2400T9	6	4	
284	2400T9	7	5	4
380	3410T9	8		1
381	3410T9	9		2
382	3420T9	10		
383	3420T9	11		

Figure 3. How the PUB Table is Scanned

X'ss' Device specifications (used to specify mode settings for 7-track and 9-track magnetic tapes). If X'ss' is not specified at system generation time or at IPL time, the system assumes X'90' for 7-track tapes and X'C0' for 9-track tapes, except for the 3420 for which X'D0' is assumed, specifying the maximum byte density for the device. C0 is the normal reset mode for a 9-track tape unit and specifies the maximum byte density for that device. C8 is an alternate mode setting for 9-track dual density tapes only. For 800 BPI single density 9-track tape, a specification of C8 reduces the time required to OPEN an output file.

The standard mode is entered in the PUB table at system generation or at IPL time. If the mode setting (different from, or the same as the standard mode) is specified in a temporary ASSGN statement, it becomes the current mode setting and is entered as such in the PUB table. When the current job ends, the standard mode is restored in the PUB table. The mode specification in a permanent ASSGN becomes the standard mode. If the X'ss' parameter is not specified for a job, the mode is the same as the standard mode.

The specifications are:

ss	Bytes per Inch	Parity	Translate Feature	Convert Feature
10	200	odd	off	on
20	200	even	off	off
28	200	even	on	off
30	200	odd	off	off
38	200	odd	on	off
50	556	odd	off	on
60	556	even	off	off
68	556	even	on	off
70	556	odd	off	off
78	556	odd	on	off
90	800	odd	off	on
A0	800	even	off	off
A8	800	even	on	off
B0	800	odd	off	off
B8	800	odd	on	off
C0	1600	single density 9-track		
C0	1600	dual density 9-track		
C8	800	single density 9-track		
C8	800	dual density 9-track		
D0	6250	single density 9-track		
D0	6250	dual density 9-track		

Note: The first 15 entries in this table are valid only for 7-track tape. The last six entries are valid only for 9-track tape.

In order to read a 7-track tape backwards, you must first create the tape file with the data convert feature off.

Also, under certain conditions, you are responsible for setting the mode of the tape to be processed. When using PIOCS with dual density tape units, a mode set must be issued if a mode is desired other than the one in which the tape was previously written. You should position the tape at LOAD POINT and issue a SET MODE command, followed by a WRITE command.

ALT

Indicates an alternate magnetic tape unit that is used when the capacity of the original assignment is reached. The specifications for the alternate unit are the same as those of the original unit. The characteristics of the alternate unit must be the same as those of the original unit. The original assignment and an alternate assignment must both be permanent or temporary assignments. Multiple alternates can be assigned to a symbolic unit.

When SYSIPT is assigned to a magnetic tape device, the file may not be multivolume.

If the original unit is reassigned, the alternate unit must also be reassigned. The ALT operand is invalid for SYSRDR, SYSIPT, SYSIN, SYSLNK, SYSCLB, and SYSLOG.

H1

Indicates that input hopper one will be used for input on the 2560 or 5425. If neither H1 nor H2 is specified, H1 is assumed.

H2

Indicates that input hopper two will be used for input on the 2560 or 5425. Note that hopper specifications are significant only for device independent files associated with the logical units SYSIPT, SYSRDR, SYSIN, and SYSPCH. In all other cases they are ignored. If both hoppers are used, they must be assigned to the same partition.

PERM
TEMP

Indicates whether the assignment should be permanent (PERM) or temporary (TEMP). It is thus possible to override the // specification or omission.

VOL=volserno

Specifies the volume serial number of the device required. This option may be specified only for tapes and disks.

If VOL is specified, the system searches for the first free unit in the requested sequence and checks the volume label to see if the required disk or tape is mounted. If not, the next free unit is checked, and so on until the proper volume serial number is found or until the end of the specified sequence is reached. The requested volume must be mounted on the unit specified in the message 1T50A MOUNT nnnnnn ON X'cuu'

Note: In a mixed device configuration, specification of TAPE,VOL or DISK,VOL may cause the system to issue a request for a volume to be mounted on the first device that becomes available. Thus, the system may request a 9-track tape to be mounted on a device that can only accommodate 7-track tapes. Likewise, a request may be issued for a 2316 disk pack to be mounted on a 3330 or 3340.

SHR This option can be specified only for disk devices and has meaning only in combination with address-list, device-class, and device-type (see corresponding discussions). If the option is not specified, the system assigns the unit to a private disk device. Therefore, unless a private device is required, it is recommended to use the SHR operand in combination with generic assignments.

BATCH

The BATCH command (Start or Continue Processing) serves to start or continue processing in one of the foreground partitions or to continue processing in the background partition. The function of the BATCH command is exactly the same as that of the START command. If the specified partition is available, job control reads the operator's next command from SYSLOG. When the operator desires to give control to another command input device, he makes an assignment to SYSRDR or SYSIN, and presses the END or ENTER key.

If the specified partition has been temporarily halted by a STOP command, it is made active. If the partition is in operation, it continues, and message

1P1nD AREA NOT AVAILABLE

is issued to the operator. In either instance, attention routine communication with the operator terminates following the BATCH command.

AR Format

$$\text{BATCH}\begin{Bmatrix}\underline{\text{BG}}\\ \text{F1}\\ \text{F2}\\ \text{F3}\\ \text{F4}\end{Bmatrix}$$

If the operand is omitted, BG is assumed.

CANCEL

The CANCEL command (Cancel Job), when used as a job control command, cancels the execution of the current job in the partition in which the command is given.

When used as an attention command with an operand specified, it cancels the execution of the current job in the specified partition.

JCC Format

CANCEL

AR Format

$$\text{CANCEL}\begin{Bmatrix}\text{BG}\\ \text{F1}\\ \text{F2}\\ \text{F3}\\ \text{F4}\end{Bmatrix}$$

BG Indicates that the background job is to be canceled.

F1-F4 Indicates that the foreground-one, -two, -three, or -four job is to be canceled.

In a one-partition system the default is BG. In multiple-partition systems the operand must be specified. The CANCEL command cannot be used to cancel the partition in which POWER/VS is running, nor the one in which VTAM is running.

CLOSE

The CLOSE command (Close Output Unit) is used to close either a system or programmer output logical unit assigned to a magnetic tape, or a system logical unit assigned to a disk or 3540 Diskette.

The CLOSE statement is used to close either a system or programmer logical unit assigned to tape. It applies only to temporarily assigned logical units.

The logical unit can optionally be reassigned to another device, unassigned, or, in the case of a magnetic tape file, switched to an alternate unit. When SYSxxx is a system logical unit (SYSLST, SYSPCH, etc.), one of the optional parameters must be specified. When closing a programmer logical unit (SYS000-SYSnnn), no optional parameter need be specified. When none is specified, the programmer logical unit is closed and the assignment remains unchanged.

Closing a magnetic tape unit consists of writing a tapemark, an EOV trailer record, two tapemarks, and rewinding and unloading the tape. The trailer record contains no block count, and later access by logical IOCS may result in a 4131D message, which can be ignored.

JCS Format

```
               ┌,X'cuu'[,X'ss']┐
// CLOSE SYSxxx│,UA            │
               │,IGN           │
               │,ALT           │
               └,SYSyyy        ┘
```

JCC Format

```
            ┌,X'cuu'[,X'ss']┐
CLOSE SYSxxx│,UA            │
            │,IGN           │
            │,ALT           │
            └,SYSyyy        ┘
```

SYSxxx For the CLOSE command only: For disk or 3540 Diskette: SYSIN, SYSRDR, SYSIPT, SYSPCH, or SYSLST

For both the statement and the command: For magnetic tape: SYSPCH, SYSLST, SYSOUT, or SYS000-SYSnnn

X'cuu' Specifies that after the logical unit is closed, it will be assigned to the channel and unit specified. c is the channel number (0-6) and uu is the unit number 00-FE (0-254) in hexadecimal. In the case of a system logical unit, the new unit will be opened if it is either a disk, 3540 Diskette, or a magnetic tape at load point.

X'ss' Device specification for mode settings on 7-track and 9-track tape. The specifications are shown in ASSGN -- Assign Logical Name. If X'ss' is not specified, the mode settings remain unchanged. The LISTIO command may be used to determine the current mode settings for all magnetic tape units.

UA Specifies that the logical unit is to be closed and unassigned.

IGN Specifies that the logical unit is to be closed and unassigned with the ignore option. This operand is invalid for SYSRDR, SYSIPT, or SYSIN.

ALT Specifies that the logical unit is to be closed and an alternate unit is to be opened and used. This operand is valid only for system output logical units (SYSPCH, SYSLST, or SYSOUT) currently assigned to a magnetic tape unit.

SYSyyy Specifies that after SYSxxx is closed, it will be assigned to the physical device to which SYSyyy iscurrently assigned (and to whicn it remains assigned). If SYSyyy is a system logical unit, it will be opened if it is either a disk, 3540 diskette, or a magnetic tape at load point.

DATE

The DATE statement contains the job date that is put in the communication region. It is in either of the two following formats:

JCS Formats

```
// DATE mm/dd/yy
// DATE dd/mm/yy
```

mm = month (01 to 12)
dd = day (01 to 31)
yy = year (00 to 99)

When the DATE statement is used, it applies only to the current job being executed, except for DASD output file labels for which the date from the SET command is used. Job control does not check the operand except for a length of eight characters. If no DATE statement is used, job control supplies the date given in the last SET command.

DLAB

The DLAB statement (DASD label information), completed in a continuation statement, contains file label information for DASD label checking and creation. This statement must immediately follow a volume (VOL) statement. The combination of this statement with VOL and XTENT should not be used to provide the extent and label information for a 3330, 3333, or 3340. For a detailed discussion of DLAB, see DOS/VS DASD Labels, GC33-5375. The DLAB statement cannot be used for VSAM files.

JCS Format

```
// DLAB 'label fields 1-3',        C
      xxxx,yyddd,yyddd,'systemcode'[,type]
```

'label fields 1-3'
The first three fields of the Format 1 DASD file label are contained just as they appear in the label. This is a 51-byte character string, contained within apostrophes and followed by a comma. The entire 51-byte field must be contained in the first of the two statements. Column 72 must contain a continuation character. The columns between

the comma and the continuation character must be blank. The Format 1 label is shown in "Appendix A". Fields 1-3 are:

File Name. 44-byte alphameric including file ID and, if used, generation number and version number of generation.

Format Identifier. 1-byte, EBCDIC 1.

File Serial Number. 6-byte alphameric, must be the same as the volume serial number in the volume label of the first or only pack of the file.

C
Continuation punch in column 72.

xxxx
Volume Sequence Number. This 4-digit EBCDIC number is the EBCDIC equivalent of the 2-byte binary volume sequence number in field 4 of the Format 1 label. This number must begin in column 16 of the continuation statement. Columns 1-15 are blank.

yyddd,yyddd
The file creation date, followed by the file expiration date. These two 5-digit numbers are the EBCDIC equivalent of the 3-byte discontinuous binary dates in fields 5 and 6 of the Format 1 label. yy is the year (00-99), and ddd is the day of the year (001-366).

'systemcode'
System code is a 13-character string, within apostrophes. For an output file, it is written in field 8 of the Format 1 label. It is ignored when used for an input file. This field is not used by the DOS/VS label processing routines. It is recommended that this field be left blank.

type
This is a two- or three-character field indicating the type of file, as follows:

SD	for sequential disk or for DTFPH with MOUNTED=SINGLE
DA	for direct access or for DTFPH with MOUNTED=ALL
ISC	for indexed sequential using load create
ISE	for indexed sequential using load extension, add, or retrieve

If this operand is omitted, SD is assumed.

DLBL

The DLBL statement (DASD label information) replaces the VOL and DLAB statement combination used in previous versions of the system. It contains file label information for DASD or 3540 Diskette label checking and creation. (Programming support for the previous VOL, DLAB, and XTENT combinations will be continued.) Together with the EXTENT statement, only DLBL should be used to supply extent and label information for the 3330/3333, 3340,or 3540. For a detailed discussion of DLBL see DOS/VS DASD Labels, GC33-5375.

JCS Format

```
// DLBL filename,['file-ID'],
   [date],[codes],[DSF],[BUFSP=n],
   [CAT=filename]
```

Continuation statements or commands are supported for DLBL.

A comma must be inserted for each positional parameter that is omitted, if any of the subsequent operands are used.

filename
This can be from one to seven alphameric characters, the first of which must be alphabetic. This unique filename is identical to the symbolic name of the program DTF that identifies the file. For VSAM, filename is identical to (1) dname of the FILE (dname) parameter in an Access Method Services command, and (2) the DDNAME=filename parameter of the Access-Method Control Block (ACB) in the processing program that identifies the file. If the DDNAME parameter is omitted, the filename must be placed in the symbolic name (label) field of the ACB.

'file-ID'
The unique name associated with the file on the volume. This can be from one to 44 bytes of alphameric data, contained within apostrophes, including file-ID and, if used, generation number and version number of generation. If fewer than 44 characters are used, the field is left-justified and padded with blanks. If this operand is omitted, filename is used. The 3540 Diskette uses a maximum of eight characters in file-ID. For VSAM, file-ID must be specified when an existing (input) file is being processed.

The file-ID is identical to the name of the file, specified in the DEFINE command of Access Method Services, listed in the VSAM catalog. When a new VSAM data space or file is being created (defined), the file-ID is ignored if it is specified. For VSAM, the file-ID must be coded according to the following rules:

- One to 44 alphameric (A-Z, 0-9, @, $, or #) characters or hyphen (-) or plus zero (+0) enclosed in apostrophes (').
- After each group of eight or less characters, a period (.) must be inserted.
- No embedded blanks are allowed.
- The first character of the file-ID and the first character following a period must be alphabetic or notational (A-Z, @, $, #).

date

This can be from one to six characters indicating either the retention period of the file in the format d through dddd (0-9999), or the absolute expiration date of the file in the format yy/ddd (75/032). If 00/ddd is specified, ddd is treated as a retention period.

If this operand is omitted, a 7-day retention period (based on the date entered via the SET command) is assumed. If this operand is present on an input file, it is ignored. For VSAM, this parameter overrides the expiration date specified in the DEFINE command of Access Method Services. However, VSAM files or data spaces can only be deleted through the DELETE command of Access Method Services even though the expiration date has been reached.

codes

This is a two to four character field indicating the type of file label, as follows:

SD for sequential disk or for DTFPH with MOUNTED=SINGLE

DA for direct access or for DTFPH with MOUNTED=ALL

DU for 3540 diskette

ISC for indexed sequential using load create

ISE for indexed sequential using load extension, add, or retrieve

VSAM for all Virtual Storage Access Method files

If this operand is omitted, SD is assumed.

DSF

This operand indicates that a data secured file is to be created or processed. At OPEN time, if a data-secured file is accessed, a warning message is issued to the operator who then decides whether the file may be accessed.

This operand is ignored for the 3540 diskette and for VSAM. All VSAM files are data secured. The DSF operand is not required for an <u>input</u> file, and it does not invoke data security if the file was not originally created as a data secured file.

BUFSP=n

If a VSAM file is to be processed, this operand specifies the number of bytes of virtual storage (0 - 999999) to be allocated as buffer space for this file. It overrides both the BUFFERSPACE parameter of the DEFINE command of Access Method Services and the BUFSP operand in the ACB macro, if its value is higher.

CAT=filename

This operand is only valid in a DLBL statement for a VSAM file. It specifies the filename (1 to 7 alphameric characters) of the DLBL statement for the catalog owning this VSAM file. The system searches only this catalog for the file-ID when the VSAM file is to be opened. Specify this operand only if you want to override the system's assumption that the job catalog or, if there is no job catalog, that the master catalog owns the file.

In a system <u>with</u> a job catalog specify:

nothing for the job catalog
private name for a private user cata
IJSYSCT for the master catalog.

In a system <u>without</u> a job catalog specify:

nothing for the master catalog
private name for a private user catalog.

DSPLY

The DSPLY command (Display Virtual Storage) allows the operator to display 16 bytes of virtual storage, starting at the specified hexadecimal address, on the device assigned to SYSLOG. Two characters (0-9,A-F) appear on SYSLOG for each byte of information; these characters represent the hexadecimal equivalent of the current information in virtual storage.

AR Format

DSPLY address

address The six digit hexadecimal address, with leading zeros if necessary, at which the storage display starts.

If the specified address is within an invalid address area, the command is ignored and a corresponding informatory message is issued.

If the 16 bytes to be displayed cross the boundary from a valid to an invalid address space, only the bytes in the valid address space are displayed, and a corresponding informatory message is issued.

DUMP

The DUMP command allows the operator to print part or all of the real and/or virtual address area contents on SYSLST. The SYSLST used may be assigned to any partition, but it must be a printer. The printer should not, at the time of the dump, be used by the partition to which it is assigned since this could result in interspersed partition and dump output.

AR Format

```
          Dump Area Operand      SYSLST Operand

         ⎧ blank            ⎫    ⎡(BG)⎤
         ⎪ S                ⎪    ⎣(Fn)⎦
         ⎪ BG               ⎪
         ⎪ Fn               ⎪
    DUMP ⎨ BGS              ⎬
         ⎪ FnS              ⎪
         ⎪ PDAREA           ⎪
         ⎩ address,address  ⎭
```

where n = 1, 2, 3, or 4

Dump Area Operand

blank If the dump area operand is omitted, the following is printed:

- the contents of the general registers, and
- the contents of all real and virtual partitions in which programs are currently running.

In the listings produced, the contents of both real and virtual partitions are in consecutive order; invalid address spaces (page pool) are indicated.

S Same as if dump area operand was omitted. The contents of the supervisor area are also dumped.

BG
Fn The contents of the specified partition and its associated registers are printed. If a program is running in real mode in the specified partition, only the real, not the associated virtual partition, is dumped. If a program is running in virtual mode in the specified partition, the virtual partition is dumped, including any fixed pages.

BGS
FnS Same as if BG/Fn was specified. The contents of the supervisor area are also dumped.

PDAREA The contents of the PD table, PD area, and the alternate address area, if present, are printed.

address, address Specifies the start and end addresses of the storage area to be dumped. The contents of the general registers that are associated with the specified storage area are also printed.

If the specified addresses are within an invalid address area, the command is ignored and an informatory message is issued.

If the storage area crosses the boundary from a valid to an invalid address space, only the contents of the specified valid address space are dumped, together with the general registers that are associated with the valid address area. A corresponding informatory message is issued.

If the storage area to be dumped crosses partition boundaries, the specified storage area is dumped, together with those general registers that belong to the partition in which the starting address is located. To obtain a dump of the SVA, refer to DOS/VS Serviceability Aids and Debugging Procedures, GC33-5380.

SYSLST Operand

BG
Fn — The operand indicates the partition to which SYSLST is assigned for output of the dump. If it is omitted, the SYSLST assigned to the background partition is used.

DVCDN

The DVCDN command (Device Down) informs the system that a device is no longer physically available for system operations. If any temporary or standard assignment were made to the device specified in the command, they are unassigned when the command is accepted.

Note: The assignment of SYSRDR is changed when a procedure is used.

If the unit is a DASD device or 3540 diskette, issue a CLOSE command for any system logical units currently assigned to it before issuing DVCDN. The DVCDN command unassigns these units without closing them. If a DVCDN command is issued with a system I/O unit assigned to the DASD device or 3540 diskette, closing the file or reassigning it to another DASD device or 3540 diskette is impossible. If an alternate assignment was made for the device specified, the alternate is removed. This command utilizes the logical transient area, and blocks out operator communicatior functions until it is completed. A DVCUP command must be issued before the device can be used again.

Note: A Device Down command is rot accepted for a device on which SYSRES, SYSREC, SYSVIS, or SYSCAT resides. However, if SYSCAT was already unassigned while being on a different device than the one that holds SYSRES, SYSREC, and SYSVIS, Device Down for the device that holds SYSCAT is accepted.

JCC Format

```
DVCDN X'cuu'
```

The entry X'cuu' is expressed in hexadecimal form, where c is the channel number (0-6) and uu is the unit number, 00-FE (0-254) in hexadecimal.

DVCUP

The DVCUP command (Device Up) informs the system that a device is available for system operations after the device has been down. As all assignments were removed by the preceding DVCDN command, an ASSGN command must be used to reassign this device.

JCC Format

```
DVCUP X'cuu'
```

The entry X'cuu' is expressed in hexadecimal form, where c is the channel number (0-6) and uu is the unit number, 00-FE (0-254) in hexadecimal.

The DVCUP command makes use of supervisor services that prevent other operator communication during execution of this command.

ENDSD

The ENDSD command (End SD Aids) is used to terminate the execution of the SDAID program. It is used in debugging procedures and described in detail in DOS/VS Serviceability Aids and Debugging Procedures, GC33-5380.

AR Format

```
ENDSD
```

END or ENTER

The END or ENTER (End of Communication) command must be issued whenever the operator has finished communicating with the system. It causes the communication routine to return control to the mainline job. END applies to CPU models without DOC (display operator console) support. ENTER applies to CPU models with DOC support.

JCC and AR Format

Press the END or ENTER key.

EXEC

The EXEC command or statement (Execute Program or Procedure) indicates either

- the end of control information for a job step and the beginning of execution of a program, in which case it must be the last command or statement processed before a job step is executed, or
- that a cataloged procedure is to be retrieved from the procedure library by job control. In this case, other commands or statements may follow EXEC.

JCS Format

```
// EXEC⎡[[PGM=]progname[],REAL][,SIZE=size]⎤
       ⎣PROC=procedurename[,OV]             ⎦
```

When issued (either from the console or from the card reader) to execute a procedure, control is given to the card reader at end-of-procedure.

JCC Format

```
EXEC⎡[[PGM=]progname][,REAL][,SIZE=size]⎤
    ⎣PROC=procedurename[,OV]            ⎦
```

When issued to execute a procedure (in this case, accepted from the console only), control is given to the console at end-of-procedure.

PGM=progname
progname
: Represents the name of the program in the core image library to be executed. The program name corresponds to the first or only phase of the program in the library. The program name can be from one to eight alphameric characters (0-9, A-Z, #, $, /, @). If the program to be executed has just been processed by the linkage editor, the program name is omitted and the PGM keyword cannot be used.

REAL
: Indicates that the job step started by EXEC will be executed in real mode. If REAL is not specified, the job step is always executed in virtual mode.

SIZE=size
: Defines how large a partition is needed for the program about to be executed. The SIZE parameter can be specified in combination with REAL (a) or without REAL (b).

(a) If specified together with REAL, it specifies the size of that part of the real partition that will be needed by the job step. The remaining part of the real partition is added to the page pool.

If the SIZE parameter is omitted and REAL is specified the whole real partition is reserved for the job step.

(b) If used without REAL, it specifies the size of that part of the virtual partition that will be directly available to the job step.

For programs (such as compilers) that dynamically allocate storage within the partition for workareas, the SIZE parameter can be used to limit the virtual storage available (and thus reduce paging activity). The remainder of the partition may be used as additional storage for other modules or data required by the program in that partition. The program obtains this additional storage by issuing GETVIS macros with the required amount of storage as a parameter; it releases the storage again by issuing FREEVIS macros.

If the SIZE parameter is omitted, the whole virtual partition is used for the job initiated with EXEC. SIZE (without REAL) must always be specified for VSAM programs, ISAM programs using the ISAM Interface Program (IIP), and for using RPS support with DASD access methods, because the space left over by the program (GETVIS area) must be large enough to accommodate VSAM and Access Method Services. If the SVA eligit VSAM phases run in the SVA, the remainder of VSAM and of Access Method Services must be accommodated in the GETVIS area. Approximate values are

270K for VSAM, 50K for VSAM buffers and control blocks and 240K for Access Method Services. Exact storage requirements can be calculated on the basis of the information contained in the book module entitled VSAM in the manual DOS/VS System Generation, GC33-5377.

The SIZE parameter can be specified in the following formats:

SIZE=nK
SIZE=AUTO (see Note)
SIZE=(AUTO,nK)

where n must be greater than zero and should be a multiple of 2 (if not, the system rounds the value up to the nearest 2K boundary).

and AUTO indicates that the program size, as calculated by the system from information in the core image directory, is to be taken as the value for SIZE. For a multiple-phase program (where the first four characters of the phase name are identical), the longest phase is used.

and (AUTO,nK) indicates that job control must take program size plus nK bytes as the value for SIZE. (If this is not a multiple of two, the value is rounded up.)

Note: Do not specify SIZE=AUTO for programs that dynamically allocate storage during execution (such as linkage editor, librarian program, and compilers).

PROC=procedurename
Represents the name of the procedure to be retrieved from the procedure library. The procedure name can be from one to eight alphameric characters.

If the procedure name begins with $$, the system substitutes a partition-related character for the second $. The character that is substituted is related to the partition in which the procedure is invoked, that is,

B for the BG partition
1 for the F1 partition
2 for the F2 partition
3 for the F3 partition
4 for the F4 partition.

The procedure corresponding to this name is then retrieved for execution.

Note that the EXEC PROC command can be entered on SYSLOG only (not on SYSRDR), whereas the EXEC PROC statement may be entered on SYSLOG or on SYSRDR.

OV Indicates that overwrite statements follow EXEC. If the EXEC PROC command was used (and entered on SYSLOG), the overwrite statements must also be entered on SYSLOG.

If the EXEC PROC statement was used and entered on SYSRDR, the overwrite statements must be entered on SYSRDR (not on SYSLOG).

If you execute a non-ending job from the procedure library, you cannot update the procedure library.

For more details, refer to the DOS/VS System Management Guide, GC33-5371.

EXTENT

The EXTENT statement (DASD Extent Information) defines each area, or extent, of a DASD file. One or more EXTENT statements must follow each DLBL statement except for single input files for sequential disk on a disk or 3540 diskette provided the DEVADDR parameter has been specified in the DTF table. For a detailed discussion of EXTENT, see DOS/VS DASD Labels, GC33-5375. Together with the DLBL statement, only EXTENT should be used to supply extent and label information for the 3330/3333, 3340, and 3540 diskette.

Note: The EXTENT cards should be checked carefully because an invalid field in the card causes the default options or the values entered by the previous EXTENT card to be overwritten by the valid entries of the flagged statement.

Multiple extent cards are valid for system files on the 3540 diskette. Valid parameters are symbolic unit, serial number, and type. The other parameters will be ignored.

This command or statement replaces the XTENT command or statement used in previous versions of the system. (Programming support for XTENT continues.)

JCS Format

```
// EXTENT [symbolic-unit],
          [serial-number],[type],
          [sequence-number],
          [relative-track],
          [number-of-tracks],
          [split-cylinder-track],
          [B=bins]
```

symbolic unit

A six-character field indicating the symbolic unit (SYSxxx) of the volume for which this extent is effective. If this operand is omitted, the symbolic unit of the preceding EXTENT, if any, is used. If this operand is omitted on the first or only EXTENT statement, the symbolic unit specified in the DTF is assumed. A symbolic unit included in the extent information for SAM, DAM, or diskette files, however, overrides the DTF DEVADDR=SYSnnn specification. (This operand is not required for an IJSYSxx filename, where xx is IN, PH, LS, LN, RS, SL, or RL, or for a file defined with the DTF DEVADDR=SYSnnn.) If SYSRDR or SYSIPT is assigned, this operand must be included. This operand is required for VSAM.

In multivolume SAM, DAM, and diskette files, each different symbolic unit must be assigned to a separate physical device. In multiextent SAM, DAM, and diskette files, all extents on one physical unit must have the same symbolic unit number. For DAM files, the extent statements must be in ascending order.

serial number

From one to six characters indicating the volume serial number of the volume for which this extent is effective. If fewer than six characters are used, the field is padded to the left with zeros.

If this operand is omitted, the volume serial number of the preceding EXTENT is used. Therefore, when a multivolume file is being processed, the volume serial number of the first volume is assumed for the entire file, unless you specify this field for the first extent of each following volume. If no serial number was provided in the EXTENT statement, the serial number is not checked and it is your responsibility if files are destroyed because the wrong volume was mounted. This operand is required for VSAM.

For the 3540 diskette, this operand specifies that the associated file will be found on this volume. If the parameter is omitted, the OPEN routines assume that the volume that was mounted is the correct one. Label checking will be done for input files and space will be allocated for an output file.

One extent card must be submitted for each volume of an input file, and sufficient extent cards must be submitted for output files to ensure that enough volumes are present to contain the file.

type

One character indicating the type of the extent, as follows:

1 - data area (no split cylinder)

2 - independent overflow area (for indexed sequential file)

4 - index area (for indexed sequential file)

8 - data area (split cylinder, for SAM files only)

If this operand is omitted, type 1 is assumed. Type 1 is the only valid parameter for diskette files.

Note: For indexed sequential files, the extent information should be entered in the following order:

1. Master index sequence number 0 and type 4
2. Cylinder index sequence number 1 and type 4
3. Prime data area sequence number 2, 3, ...,n and type 1
4. Independent overflow area sequence number (n+1) and type 2

where n is the sequence number of the last prime data area extent.

Note also, the master and the cylinder index must be in adjacent areas on the same logical unit.

sequence number
: One to three characters containing a decimal number from 0 to 255 indicating the sequence number of this extent within a multiextent file. Extent sequence 0 is used for the master index of an indexed sequential file. If the master index is not used, the first extent of an indexed sequential file has the sequence number 1. The extent sequence number for all other types of files begins with 0. If this operand is omitted for the first extent of ISAM files, the extent is not accepted. For SAM, VSAM, or DAM files, this operand is not required. Sequence checking via this parameter is ignored for diskette files. Extents are numbered on the label cylinder in the order submitted.

relative track
: One to five characters indicating the sequential number of the track, relative to zero, where the data extent is to begin. If this field is omitted on an ISAM file, the extent is not accepted. This field is not required for SAM input files (the extents from the file labels are used). This field must be specified for DAM input files.

For VSAM, this operand must be specified when a data space or a file with the UNIQUE option is being created (defined by Access Method Services). This operand is not required, and it is ignored if it is specified, when a VSAM file is created within an existing data space. In this case, the space for the file is sub-allocated by VSAM from direct-access extents it already owns. This operand is also not required for VSAM input files because the extents are obtained from the VSAM catalog.

When using split cylinder files, this parameter designates the beginning of the split as well as the first track of the file.

Formulas for converting actual to relative track (RT) and relative track to actual for the DASD devices follow.

__Actual to Relative__

2311	10 x cylinder number + track number = RT
2314/ 2319	20 x cylinder number + track number = RT
2321	1000 x subcell number + 100 x strip number + 20 x cylinder number + track number = RT
3330/ 3333	19 x cylinder number + track number = RT
3340	12 x cylinder number + track number = RT

__Relative to Actual__

2311	$\frac{RT}{10}$ = quotient is cylinder, remainder is track
2314/ 2319	$\frac{RT}{20}$ = quotient is cylinder, remainder is track
2321	$\frac{RT}{1000}$ = quotient is subcell, remainder1 $\frac{remainder1}{100}$ = quotient is strip, remainder2 $\frac{remainder2}{20}$ = quotient is cylinder, remainder is track
3330/ 3333	$\frac{RT}{19}$ = quotient is cylinder, remainder is track
3340	$\frac{RT}{12}$ = quotient is cylinder, remainder is track

__Example__: Track 5, cylinder 150 on a 3330 = 2855 in relative track.

number of tracks
: One to five characters indicating the number of tracks to be allotted to the file. For SD input, this field may be omitted. For an indexed sequential file, the number of tracks for prime data must be a multiple of 10 for 2311, of 19 for 3330 or 3333, of 12 for 3340, and of 20 for 2314 or 2319. The number of tracks for a split cylinder file must be the product of the number of cylinders for the file and the specified number of tracks per cylinder for that file.

For VSAM, this operand must be specified when a data space or a file with the UNIQUE option is being created (defined by Access Method Services). This operand is not required, and it is ignored if it is specified, when a VSAM file is created within an existing data space. In this case, the space for the file is sub-allocated by VSAM from direct-access extents it already owns. This operand is also not required for VSAM input files because the extents are obtained from the VSAM catalog.

split cylinder track
One or two characters, from 0-19, indicating the upper track number for the split cylinder in SAM files.

bins
One or two characters identifying the 2321 bin for which the extent was created, or on which the extent is currently located. If the field is one character, the creating bin is assumed to be zero. There is no need to specify a creating bin for SAM or ISAM files. If this operand is omitted, bin zero is assumed for both bins. If the operand is included and positional operands are omitted, only one comma is required preceding the key-word operand. (One comma for each omitted positional operand is acceptable, but not necessary.)

HOLD

The HOLD command is used to hold assignments if you want to stop or unbatch a partition.

JCC Format

```
     (F1[,F2][,F3][,F4])
HOLD {                 }
     (F4[,F3][,F2][,F1])
```

IGNORE

Whenever an abnormal condition arises, the operator will be notified by an appropriate message on SYSLOG. Depending on the situation, he may have to ignore the condition by entering an IGNORE command. This is indicated under 'Operator Action' in DOS/VS Messages, GC33-5379, for each message appearing on SYSLOG.

JCC and AR Format

```
IGNORE
```

JOB

The JOB statement indicates the beginning of control information for a job. It is in the following format.

JCS Format

```
// JOB jobname [accounting information]
```

jobname
The name of the job. Must be one to eight alphameric characters, except blank, equal sign (=), and comma (,). When a job is restarted, the jobname must be identical to that used when the checkpoint was taken. Any user comments can appear on the JOB statement following the jobname (through column 71). If the timer feature is present, the time of day appears in columns 73-100 when the JOB statement is printed on SYSLST. The time of day is printed in columns 1-28 on the next line of SYSLOG.

In both cases the format is

DATE mm/dd/yy,CLOCK hh/mm/ss

accounting information
If the job accounting interface has been specified during system generation, the 16 characters of user-specified accounting information are moved to the Job Accounting Table. If accounting information is specified, it must be separated from the job name by a single blank. If the job accounting interface is not specified during system generation, any information specified after the jobname is ignored.

Notes:

1. If the JOB card is omitted from the job stream, no duration and/or date is printed at end of job (when the /& card is read).

2. The start time that the job accounting routines store in byte 36 of the Job Accounting Table is calculated from the values in virtual storage locations 80 and 84 (job step start time). Step stop time for any given step is step start time for the next step.

LBLTYP

The LBLTYP statement (Reserve Storage for Label Information) defines the amount of storage to be reserved at link-edit time or at execution time (for self-relocating programs) for processing of tape and nonsequential disk file labels in the real or virtual address area of virtual storage. It applies to both background and foreground virtual programs. It is to be submitted immediately preceding the // EXEC LNKEDT statement, with the exception of self-relocating programs, for which it is instead submitted immediately preceding the // EXEC statement for the program.

Storage for VSAM label processing is reserved automatically by VSAM routines; an LBLTYP statement is not required.

JCS Format

```
// LBLTYP  TAPE[(nn)]
           NSD(nn)
```

TAPE[(nn)] Used only if tape files requiring label information are to be processed, and no nonsequential DASD files are to be processed. nn is optional, and is present only for future expansion. (It is ignored by job control.)

NSD(nn) Used if any nonsequential DASD files are to be processed regardless of other type files to be used. nn specifies the largest number of extents to be used for a single file.

The amount of storage that must be reserved for label information is:

1. For standard tape labels (any number): 80 bytes.

2. For sequential DASD and DTFPH MOUNTED=SINGLE: 0 bytes.

3. For DTFIS, DTFDA, and DTFPH MOUNTED=ALL: 84 bytes plus 20 bytes per extent.

The area reserved is that required by the file with the largest requirement. This area is used during OPEN.

LFCB

The LFCB command causes the system to load the buffer image, contained in the named core image library phase, into the forms control buffer of the specified printer. The command can be used for any DOS/VS supported printer on which forms skip operations are controlled by an FCB. During the time the printer in question is printing the output of a program this command should be used with extreme caution, as there is no way of predicting when the printer will be finished printing the output under control of the current FCB load. For a printer in operation it is recommended that the operator use this command if, for example, printing the output for a particular program started under control of the wrong FCB load and he is able to correct this by issuing the command. When the program runs in a POWER/VS-controlled partition, this command can be used with the restart capability of POWER/VS.

AR Format

```
LFCB     X'cuu',phasename[,FORMS=xxxx]
        [,LPI=n][,NULMSG]
```

X'cuu' specifies the channel and unit number of the printer whose FCB is to be loaded.

Phasename specifies the name of the core image library phase that contains the applicable buffer load image.For detailed information on the contents and format of this phase refer to the section "System Buffer Load (SYSBUFLD)".

FORMS=xxxx specifies the installation-defined forms number xxxx of the paper that is to be used with the new FCB load. For xxxx, substitute from one to four alphameric characters. If the new FCB load requires a change of forms, this operand must be specified to ensure proper system operation.

LPI=n (for a printer other than a 3211). Indicates the required setting of the carriage clutch. For n, you can either substitute 6 (six lines per inch) or 8 (eight lines per inch). The system inserts this operand in a buffer load action message.

NULMSG specifies that the printing of a buffer load verification message is to be suppressed. If NULMSG is specified, the system continues processing immediately after the FCB load operation has been completed, and the operator is unable to verify that the contents of the FCB match the forms to be used.

LISTIO

The LISTIO command or statement (List I/O Assignment) causes the system to print a listing of I/Q assignments. The listing appears on a SYSLOG (command) or SYSLST (statement). If SYSLST is not assigned, the LISTIO statement is ignored.

JCS Format

```
               ⎧ SYS    ⎫
               ⎪ PROG   ⎪
               ⎪ BG     ⎪
               ⎪ F1     ⎪
               ⎪ F2     ⎪
   // LISTIO  ⎨ F3     ⎬
               ⎪ F4     ⎪
               ⎪ ALL    ⎪
               ⎪ SYSxxx ⎪
               ⎪ UNITS  ⎪
               ⎪ DOWN   ⎪
               ⎪ UA     ⎪
               ⎩ X'cuu' ⎭
```

JCC Format

```
               ⎧ SYS    ⎫
               ⎪ PROG   ⎪
               ⎪ BG     ⎪
               ⎪ F1     ⎪
               ⎪ F2     ⎪
      LISTIO  ⎨ F3     ⎬
               ⎪ F4     ⎪
               ⎪ ALL    ⎪
               ⎪ SYSxxx ⎪
               ⎪ UNITS  ⎪
               ⎪ DOWN   ⎪
               ⎪ UA     ⎪
               ⎩ X'cuu' ⎭
```

SYS — Lists the physical units assigned to all background system logical units or the physical units assigned to all system logical units of the particular foreground partition.

PROG — Lists the physical units assigned to all background programmer logical units.

BG — Lists the physical units assigned to all logical units of the background partition.

F1-F4 — Lists the physical units assigned to all logical units of the particular foreground partition.

ALL — Lists the physical units assigned to all logical units.

SYSxxx — Lists the physical units assigned to the logical unit specified (invalid for SYSOUT and SYSIN). The assignment is given for the partition from which the command is given.

UNITS — Lists the logical units assigned to all physical units.

DOWN — Lists all physical units specified as inoperative.

UA — Lists all physical units not currently assigned to a logical unit.

X'cuu' — Lists the logical units assigned to the physical unit specified.

Physical units are listed with current device specification for magnetic tape units. Logical units are listed with ownership (background, foreground-one, -two, -three, or -four), when applicable. If a unit has a standard assignment in one mode and a temporary assignment in another mode, the CMNT column identifies the type of assignment for each indicated mode. An example of a listing produced by the LISTIO F1 command is shown in Figure 4. All channel and unit numbers are represented in hexadecimal.

If a standard or permanent assignment is temporarily overwritten, STD in the CMNT (comment) column indicates the permanent assignment, whereas the temporary assignment is printed one line above (see, for example, SYSSLB in Figure 4).

If a device is assigned only temporarily, UA is printed in the STD line (see SYS002 in Figure 4).

ALT in the CMNT column indicates that an alternate tape has been assigned.

LOG

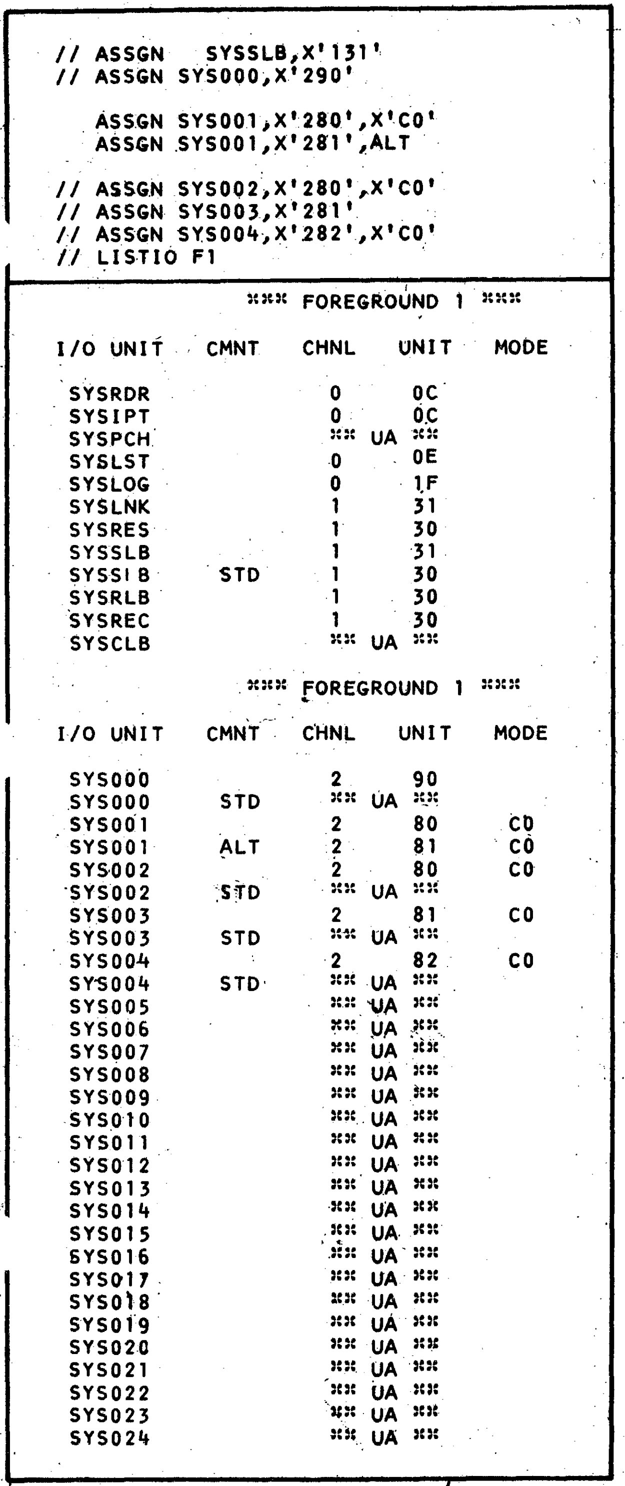
The LOG command causes the system to log, on SYSLOG, columns 1-72 of all job control commands and statements occuring in the partition in which the LOG is issued. The AR LOG affects all the partitions. The LOG function is effective until a NOLOG command for the partition involved is sensed.

The LOG command suppresses the ACANCEL function in the supervisor.

JCC and AR Format

```
LOG
```

The operand field is ignored by the system.



```
// ASSGN   SYSSLB,X'131'
// ASSGN SYS000,X'290'

   ASSGN SYS001,X'280',X'C0'
   ASSGN SYS001,X'281',ALT

// ASSGN SYS002,X'280',X'C0'
// ASSGN SYS003,X'281'
// ASSGN SYS004,X'282',X'C0'
// LISTIO F1

               *** FOREGROUND 1 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYSRDR             0       0C
 SYSIPT             0       0C
 SYSPCH             ** UA  **
 SYSLST             0       0E
 SYSLOG             0       1F
 SYSLNK             1       31
 SYSRES             1       30
 SYSSLB             1       31
 SYSSI B    STD     1       30
 SYSRLB             1       30
 SYSREC             1       30
 SYSCLB             ** UA  **

               *** FOREGROUND 1 ***

I/O UNIT   CMNT    CHNL    UNIT    MODE

 SYS000             2       90
 SYS000     STD     ** UA  **
 SYS001             2       80      C0
 SYS001     ALT     2       81      C0
 SYS002             2       80      C0
 SYS002     STD     ** UA  **
 SYS003             2       81      C0
 SYS003     STD     ** UA  **
 SYS004             2       82      C0
 SYS004     STD     ** UA  **
 SYS005             ** UA  **
 SYS006             ** UA  **
 SYS007             ** UA  **
 SYS008             ** UA  **
 SYS009             ** UA  **
 SYS010             ** UA  **
 SYS011             ** UA  **
 SYS012             ** UA  **
 SYS013             ** UA  **
 SYS014             ** UA  **
 SYS015             ** UA  **
 SYS016             ** UA  **
 SYS017             ** UA  **
 SYS018             ** UA  **
 SYS019             ** UA  **
 SYS020             ** UA  **
 SYS021             ** UA  **
 SYS022             ** UA  **
 SYS023             ** UA  **
 SYS024             ** UA  **
```

igure 4. Example of LISTIO F1 output

LUCB

The LUCB command causes the system to load the buffer image, contained in the named core image library phase, into the universal character set buffer (UCB) of the specified printer. The command can be used for any DOS/VS supported printer equipped with the UCS feature. During the time the printer in question is printing the output of a program this command should be used with extreme caution, as there is no way of predicting when the printer will be finished printing the output under control of the current UCB load. For a printer in operation it is recommended that the operator use this command if, for example, printing the output for a particular program started under control of the wrong UCB load and the operator is able to correct this condition by issuing the command. When the program runs in a POWER/VS-controlled partition, this command can be used because of the restart capability of POWER/VS.

AR Format

```
LUCB    X'cuu',phasename[,FOLD][,NOCHK]
        [,TRAIN=xxxxxx][,NULMSG]
```

X'cuu' specifies the channel and unit number of the printer whose UCB is to be loaded.

phasename specifies the name of the core image library phase that contains the applicable buffer load image. For detailed information on the contents and format of this phase refer to "User-Written UCB Load Phase" in the section entitled "System Buffer Load (SYSBUFLD)".

FOLD causes lower case characters to be printed as upper case characters.

NOCHK Causes data checks resulting from mismatches between printline characters and the UCB to be suppressed.

TRAIN=xxxxxx
Indicates that the print train (print chain for a 1403) identified by xxxxxx is to be mounted on the printer. The system inserts this operand in an action message. The train (or chain) identification xxxxxx may be from one to six characters in length. If a new train (or chain) must be installed, this operand is required to ensure proper system operation.

NULMSG Specifies that the printing of a buffer load verification message is to be suppressed. If NULMSG is specified, the system continues normal processing immediately after the UCB load operation has been completed and the operator is unable to verify that the contents of the UCB match the print train (or chain) mounted on the printer.

MAP

The MAP command produces on SYSLOG a map of the storage areas allocated to programs. It indicates program execution mode (R,V), and whether a partition is inactive (I) or active (A) for each partition.

JCC and AR Format

MAP

The map of storage areas allocated to partitions is produced in the following format:

AREA	K-REAL	UPPER LIMIT	K-VIRT	UPPER LIMIT	NAME
SP	92K	94207		262144	
BG V5A	30K	124927	68K	331775	MINE
F4 I4	10K	135167	80K	413695	
F3 V3D	10K	145407	80K	495615	
F2 I2	14K	159743	100K	598015	
F1 I1	14K	174079	120K	720895	
SVA			64K	786431	
VIS			0K	786431	
PP	86K	262143			

Figure 5. Map of Storage Areas

Note: The output does not indicate storage temporarily added to the page pool as a result of using the SIZE parameter of the EXEC statement.

AREA
SP = supervisor
BG, F4, F3,'F2, F1 = partition
SVA = shared virtual area
VIS = system GETVIS area
PP = main page pool
R = real mode
V = virtual mode
I = inactive partition
A = active partition (or system directory list has been built in the SVA)
D = deactivated partition (this partition has previously been active)
1-5 = priority of partition

K-REAL Shows the number of K bytes allocated in the real address area.

UPPER LIMIT Shows the highest storage address (in decimal) of each area in the real address area.

K-VIRT Shows the number of K bytes allocated in the virtual address area.

UPPER LIMIT Shows the highest storage address (in decimal) of each area in the virtual address area. For the supervisor, this field specifies the start address of the virtual address area.

NAME Contains the name of the job that is currently executing in the corresponding partition. This field is blank for the supervisor, SVA, VIS, and main page pool.

For an active partition when no JOB statement or command was entered, the name field contains NO NAME. For an inactive foreground partition, the name field is blank.

MODE

The MODE command allows you to alter the recording mode.

The MODE command provides the following options for controlling recoverable machine check interrupts (MCI):

- The mode of recording for unlabeled and nonstandard labeled tape can be reset.
- The recording mode for a particular device other than a teleprocessing device can be set to intensive or diagnostic, or no recording mode can be specified.
- The mode that the system is operating in (the status of the system) can be requested.
- An EFL threshold value can be specified to override the IBM-supplied value.
- The MODE command can also be used to place the Model 145 control storage ECC in threshold mode.

The MODE command is a notational command. Operands of the MODE command can be entered in any order and must be continuous (that is, no blanks are allowed between or within operands). The STATUS operand cannot have any other operand, before or after it.

The total length of the MODE command must not exceed 30 characters.

AR Format

```
      ⎧IR                                                 ⎫
      ⎪CR                                                 ⎪
      ⎪CE,cuu⎡,I[,xx,y]⎤                                  ⎪
      ⎪      ⎢,D[,xx,y]⎥                                  ⎪
      ⎪      ⎣,N       ⎦                                  ⎪
MODE  ⎨                                                   ⎬
      ⎪R                                                  ⎪
      ⎪STATUS                                             ⎪
      ⎪HIR    ⎡,M⎤*⎧⎡,R ⎤[,E=eeee][T=tttt]*⎫              ⎪
      ⎪ECC    ⎣,C⎦ ⎨⎢,Q ⎥                  ⎬              ⎪
      ⎩            ⎩⎣,TH⎦                  ⎭              ⎭
```

*Note: When either HIR or ECC is specified, at least one of the optional operands must be selected. TH is only valid for the Model 145 when ECC,C is specified with the MODE command.

For a Model 115 or 125, a MODE command is valid only if RMSR is supported. Even then, only the operands IR, CR, and CE may be used. The operands R, STATUS, HIR, and ECC must never be specified for a Model 115 or 125.

The meanings of the operands are:

IR
CR Recording mode for nonstandard labeled and unlabeled tape. Specify Individual Recording (IR) if you wish to record and then reset the tape error statistics at each tape OPEN. Specify Combined Recording (CR) to accumulate all the statistics from nonstandard labeled and unlabeled tape on a specific tape unit until a standard labeled tape is opened. Then one recording of the statistics from all the nonstandard labeled and unlabeled tapes is made on SYSREC, and the statistical counters are reset in the PUB2 table.

CE The recording mode for a device at physical location X'cuu' may be reset. The possible recording modes are:

Normal. The default, normal, is assumed.

I Intensive. Normal recording continues. In addition, the next seven errors of a particular type (xx,y) or the next seven errors of any type (if xx,y is not specified) are recorded. The number of I/O retries required for success is not recorded.

D Diagnostic. Normal recording continues. In addition, the next seven errors of a particular type (xx,y) or the next seven errors of any type (if xx,y is not specified) are recorded. The number of I/O retries required for success is also recorded.

N No recording.

When the recording mode parameter is the last parameter of the MODE command, a check is made to see if all errors are recorded. When in intensive or diagnostic mode, it is possible to check for only one type of error. Indicate the bit to be examined with:

(xx,y) where y is the bit (0-7) and xx the byte (0-31) of sense data to be checked.

STATUS A report is printed on SYSLOG which indicates:

- The type of facility used (HIR,ECC)
- System mode of operation
- Current error count
- Error count threshold
- Current elapsed time
- Time threshold
- Number of buffer pages deleted.

A buffer page is a 32-byte work area in control storage that is used by the Model 155 hardware program.

The status report formats are:

```
    ⎧R⎫
HIR,⎩Q⎭,aaaa/eeee,bbbb/tttt
```

For the Model 135

```
    ⎧R⎫
ECC,⎩Q⎭
```

For the Model 145

ECC,$\left\{\begin{matrix}R\\Q\end{matrix}\right\}$,$\left\{\begin{matrix}M\\C\end{matrix}\right\}$,aaaa/eeee,bbbb/tttt

where:

aaaa = Current error count
eeee = Error count threshold
bbbb = Current elapsed time
tttt = Total threshold

For the Models 155-II and 158

ECC,$\left\{\begin{matrix}R\\Q\end{matrix}\right\}$,aaaa/eeee,bbbb/tttt

BUF DLT=xxx

where:

aaaa = Current error count
eeee = Error count threshold
bbbb = Current elapsed time
tttt = Time threshold
xxx = Total number of inoperable buffer pages deleted.

HIR Hardware Instruction Retry. This operand changes the mode of the HIR facility to R or Q and/or modifies the error count threshold and/or time threshold.

Note: When HIR is placed in quiet mode, ECC, also goes into quiet mode.

ECC Error Correction Code. This operand changes the mode of the ECC facility to R or Q, and/or modifies the error count threshold and/or time threshold. ECC,R and ECC,Q are the only valid modes of diagnosis for the Model 135. If ECC is specified for a Model 145, M or C must also be specified. ECC can also place the Model 145 control storage in threshold mode.

Note: Use of the Error Correction Code (ECC) in full recording mode may cause severe system degradation. Thus, the [ECC,M/C,R] operand combination of the MODE command should only be used by the customer engineer or at his request.

R Recording Mode

MODE R - places both HIR and ECC in recording mode.

MODE HIR,R - places HIR in recording mode.

MODE ECC,R (Models 155-II and 158) - if HIR is already in recording mode, it places ECC in recording mode.

MODE ECC,M,R (Model 145) - if HIR is already in recording mode, real storage is placed in recording mode.

MODE ECC,C,R (Model 145) - if HIR is already in recording mode, control storage is placed in recording mode.

Q Quiet Mode

MODE HIR,Q - places both HIR and ECC in quiet mode.

MODE ECC,Q (Models 135, 155-II, and 158) places ECC in quiet mode.

MODE ECC,M,Q, (Model 145) - places real storage in quiet mode.

MODE ECC,C,Q (Model 145) - places control storage in quiet mode.

M or C Real or control storage: M or C is only valid for the Model 145. M or C must be specified when ECC is specified for the Model 145. M indicates real storage and C control storage.

TH Threshold Mode: on the next occurrence of an ECC control storage error, control storage is placed in quiet mode. TH is only valid for the Model 145 if ECC,C is specified. TH places the Model 145 control storage ECC in threshold mode.

E=eeee
T=tttt Values entered for E and T must be within the following decimal ranges:

E - 8 (initial value) through 9999

T - 8 (initial value) through 9999

The IBM-supplied value is 8.

Note: Whenever HIR is in quiet mode, ECC mode must not be changed.

For the Model 135, the only valid mode commands are:

MODE CE,...
MODE STATUS
MODE ECC,Q
MODE ECC,R

MSG

The MSG command transfers control to an operator communications routine previously activated by a STXIT command.

AR Format

```
MSG {BG}
    {Fn}
```

Fn indicates the desired foreground partition. BG indicates the background partition; pressing the INTERRUPT key, however, produces the same result for BG.

If the program in the specified partition has not established operator communication linkage, a message is printed on SYSLOG informing the operator of this condition.

MTC

The MTC command or statement controls IBM 2400/3400 series magnetic tape operations. The first operand specifies the operation to be performed.

JCS Format

```
// MTC opcode,SYSxxx[,nn]
```

JCC Format

```
MTC opcode,{X'cuu'}[,nn]
           {SYSxxx}
```

The first operand can be:

Opcode	Meaning
BSF	Backspace one file so tape is positioned for reading the tapemark preceding the file backspaced.
BSR	Backspace record.
DSE	Data security erase.*
ERG	Erase gap (write blank tape).
FSF	The tape is positioned beyond the tapemark following the file spaced over.
FSR	Forward space record.
RUN	Rewind and unload tape.
REW	Rewind tape.
WTM	Write tapemark.

* Data security erase (3400-series only). This command erases a tape from the point at which the tape is positioned when the operation is initiated up to the end-of-tape reflective marker. If data is written after the end-of-tape reflective marker, the data must be erased with [//] MTC ERG SYSnnn.

To ensure that a DSE failure is detected quickly, rewind or rewind-unload should be performed with an MTC rather than manually. If the DSE command is issued when the tape is at load point, the contents of the tape, including the volume label, are erased completely. In such a case the tape must be reinitialized or a tapemark must be written on it before it can be used again.

The partition that issued the [//] MTC DSE command is placed in the wait state until the end-of-tape reflective marker is reached.
The second operand, SYSxxx, represents any assigned logical unit.

X'cuu' is the channel and unit in hexadecimal where c is the channel number (0-6) and uu is the unit number, 00-FE (0-254). X'cuu' is not valid for JCS.

The optional third entry, nn, is a decimal number (01-99) representing the number of times the specified operation is to be performed. If nn is not specified, the operation is performed once.

NEWVOL

If an assignment specifying VOL= was given for a disk or tape unit and the system cannot find the requested volume on that unit, then the system prints a message on SYSLOG. The message 1T50A requests the operator to mount the desired volume. The partition enters the wait state. After having mounted the proper volume, the operator issues the NEWVOL attention command to indicate that processing may continue with the new volume.

AR Format

```
NEWVOL {BG}
       {Fn}
```

If no operand is specified, BG is assumed. Fn can be specified as F1, F2, F3, or F4. If the specified partition is not waiting for a volume to be mounted, an error message is printed on SYSLOG.

NOLOG

The NOLOG command (Suppress Logging) terminates the listing, on SYSLOG, of job control commands and statements (except JOB, PAUSE, STOP, ALLOC, ALLOCR, MAP, HOLD, DVCDN, DVCUP, *, EOJ (/&), and EOP (/+)) that occur in the partition in which the NOLOG is issued. The NOLOG attention command affects all the partitions. The NOLOG function is effective until a LOG command for the partition involved is sensed.

JCC and AR Format

NOLOG

The operand field is ignored by the system.

OPTION

The OPTION statement specifies one or more of the job control options. The format of the OPTION statement is:

JCS Format

```
// OPTION option1[,option2...]
```

The options that can appear in the operand field follow. Selected options can be in any order. Options are reset to the standards established at system generation time upon encountering a JOB or a /& statement.

LOG Lists columns 1-80 of all control statements and commands on SYSLIST. Control statements and commands are not listed until a LOG option is encountered. Once a LOG option statement is read, logging continues from job step to job step until a NOLOG option is encountered or until either the JOB or /& control statement is encountered.

NOLOG Suppresses the listing of all valid control statements and commands on SYSLST until a LOG option is encountered. If SYSLST is assigned, invalid statements and commands are listed.

DUMP Dumps the registers, supervisor area, and the virtual or the temporary real partition on SYSLST, if assigned, in the case of an abnormal program end (such as program check).

Note: If SYSLST is assigned to a 3211 printer, the indexing feature of the device must be used with care. Shifting the print line to the left or too far to the right causes characters to be left out from every printed line of the dump.

PARTDUMP Dumps the registers, the addresses and contents of the partition COMREG, the PUB table, the PUB owner, the partition LUB table, the JIB table, the partition DIB table, the label type and the partition in which the program resides. The dump will be recorded on SYSLST (if assigned) in case of an abnormal program end. Refer to the Note under DUMP.

NODUMP Suppresses the DUMP or PARTDUMP option.

LINK Indicates that the object module is to be link-edited. When the LINK option is used, the output of the language translators is written on SYSLNK. The LINK option must always precede an EXEC LNKEDT statement in the input stream. (CATAL also causes the LINK option to be set.) LINK is accepted by job control operating in a foreground partition if NPARTS=2-5 and PCIL=YES is specified during system generation and a private core image library is assigned. The LINK option is reset to NOLINK upon encountering an ASSGN statement or command for SYSLNK.

NOLINK Suppresses the LINK option. The language translators can also suppress the LINK option if the problem program contains an error that would preclude the successful execution of the problem program.

DECK Language translators produce object modules on SYSPCH. If LINK is specified, the DECK option is accepted by the PL/I, FORTRAN IV, American National Standard and DOS/VS COBOL compilers, and the assembler.

NODECK Suppresses the DECK option.

EDECK The assembler punches all valid source macro definitions in edited format on SYSPCH. These macro definitions can be included in sublibrary E of the source statement library.

NOEDECK Suppresses the EDECK option.

ALIGN The assembler aligns constants and data areas on proper boundaries and checks the alignment of addresses used in machine instructions.

NOALIGN Suppresses the ALIGN option.

LIST Language translators write the source module listing on SYSLST. The assembler also writes the hexadecimal object module listing and the assembler and FORTRAN write a summary of all errors in the source program. All are written on SYSLST.

NOLIST Suppresses the LIST option. This option overrides the printing of the external symbol dictionary, relocation list dictionary, and cross reference (XREF) list.

LISTX The ANS and DOS/VS COBOL compilers produce a PROCEDURE DIVISION MAP on SYSLST. The PL/I and FORTRAN compilers produce the object modules on SYSLST.

NOLISTX Suppresses the LISTX option.

SYM The American National Standard and DOS/VS COBOL compilers produce a DATA DIVISION map on SYSLST; the PL/I compiler produces the symbol table on SYSLST; the COBOL compilers produce a DATA DIVISION map on SYSLST.

NOSYM Suppresses the SYM option.

XREF The assembler writes the symbolic cross-reference list on SYSLST.

NOXREF Suppresses the XREF option.

ERRS The FORTRAN, ANS and DOS/VS COBOL, and PL/I compilers summarize all errors in the source program on SYSLST.

NOERRS Suppresses the ERRS option.

ACANCEL This option indicates that the job must be canceled (instead of awaiting operator intervention) if an attempt to assign a device is unsuccessful. This may be due to an undefined device, invalid device status, unassignable unit, or conflicting I/O assignments. At end-of-job, this option is set to the standard setting specified at system generation time.

NOACANCEL Suppresses the ACANCEL option. The system awaits operator intervention in the case of an unsuccessful assignment.

CATAL A phase or program is cataloged in the core image library at the completion of a link-edit run. CATAL also sets the LINK option. CATAL is accepted by job control operating in a foreground partition if NPARTS=2-5 and PCIL=YES is specified during system generation and a private core image library is assigned.

STDLABEL All DASD and tape labels submitted after this point are written at the beginning of the standard label track. Reset to USRLABEL option at end of job or end-of-job step. All file definition statements submitted after this option are available to any program in any partition until another set of standard file definition statements is submitted. STDLABEL is not accepted by job control operating in a foreground partition. All file definition statements following OPTION STDLABEL are included in the system standard file definition set until one of the following occurs:

1. End-of-job.
2. OPTION USRLABEL is specified.
3. OPTION PARSTD is specified.

OPTION STDLABEL followed by a /& clears the standard label track. (See Note.)

USRLABEL All DASD and tape labels submitted after this point are written at the beginning of the user label track. (See Note.)

PARSTD All DASD and tape labels submitted after this point are written at the beginning of the partition standard label track. Reset to USRLABEL option at end of job or end-of-job step. All file definition statements submitted after this option are available to any program in the current partition until another set of partition standard file definition statements is submitted. All file definition statements submitted after OPTION PARSTD are included in the partition standard file definition set until one of the following occurs:

1. End-of-job.
2. OPTION USRLABEL is specified.
3. OPTION STDLABEL is specified.

OPTION PARSTD followed by a /& clears the partition standard label track.

For a given filename, the sequence of search for label information during an OPEN is the USRLABEL area, followed by the PARSTD area, followed by the STDLABEL area. (See Note.)

48C Specifies the 48-character set on SYSIPT (for PL/I).

60C Specifies the 60-character set on SYSIPT (for PL/I).

SYSPARM= 'string'
Specifies a value for the assembler system variable symbol $SYSPARM. $SYSPARM gets the value of the string, which is enclosed by apostrophes. The string can contain 0-8 EBCDIC characters. One internal quote must be represented by two. (Job control removes one of them when setting the value.) The surrounding quotes are not included and the length of $SYSPARM is determined by the resulting string.

This operand is invalid if SYSPARM support was not specified when the system was originally generated.

SUBLIB=DF Directs the DOS/VS assembler and ESERV program to retrieve non-edited macros and copy-books from the D.sublibrary of the source statement library instead of from the A.sublibrary and to retrieve edited macros from the F.sublibrary instead of from the E.sublibrary. IBM uses the D. and F.sublibraries to distribute macros and copy source code for programs that are to be executed in a teleprocessing network control unit. The option remains in force until an end-of-job condition or a // OPTION SUBLIB=AE statement is encountered.

SUBLIB=AE Redirects the Assembler and the ESERV program to retrieve non-edited macros and copy books from the A.sublibrary and to retrieve edited macros from the E.sublibrary.

The options specified in the OPTION statement remain in effect until a contrary option is encountered or until a JOB control statement is read. In the latter case, the options are reset to the standard that was established when the system was originally generated.

Any assignment for SYSLNK after the occurrence of the OPTION statement cancels the LINK and CATAL options. These two options are also canceled after each occurrence of an EXEC statement with a blank operand.

Note: Refer to DOS/VS Tape Labels, GC33-5374, and DOS/VS DASD Labels, GC33-5375 for additional information about STDLABEL, USRLABEL, and PARSTD.

OVEND

The OVEND (Overwrite End) statement or command applies to cataloged procedures only. It is used to indicate that no more overwrite statements will follow for the respective procedure.

JCS Format

```
// OVEND     [comment]
```

JCC Format

```
OVEND     [comment]
```

For the use of overwrite statements and the rules that apply to temporary procedure modification, refer to the DOS/VS System Management Guide, GC33-5371.

PAUSE

The PAUSE statement causes a pause immediately after processing this statement.

The PAUSE command causes a pause at the end of the current job step.

The PAUSE statement or command always appears on SYSLOG. If only a printer is available, the PAUSE statement or command is ignored. At the time SYSLOG is unlocked for message input, you can continue processing by pressing the END or ENTER key.

JCS Format

```
// PAUSE [any user comment]
```

JCC Format

```
PAUSE [any user comment]
```

AR Format

$$\text{PAUSE}\ \left\{\begin{matrix}\underline{\text{BG}}\\ \text{F1}\\ \text{F2}\\ \text{F3}\\ \text{F4}\end{matrix}\right\}\ [\text{,EOJ}]$$

For the attention routine, if the first operand is omitted, BG is assumed. The EOJ operand must be preceded by a BG, F1, F2, F3, or F4 operand.

The attention routine PAUSE command normally causes job control processing to pause at the end of the current job step in the partition specified. Use of the EOJ operand causes job control processing to pause at end-of-job in the partition specified.

PRTY

The PRTY (priority) command allows the operator (a) to display the priority sequence of the partitions in the system, which was set at system generation, or (b) to change that sequence. In both cases also the current status (if active) of the TP Balancing function is displayed.

AR Format

PRTY

This format of the PRTY command displays the priority of all partitions on the device assigned to SYSLOG.

PRTY [p1...,pn]

This format of the command changes the priority of the partitions. The number of operands specified must coincide with the number of partitions in the system. The values of the operands are the partition identifiers BG, F1, F2, F3, or F4. The order of operands indicates the new priority sequence, in ascending order. For example

```
PRTY    BG,F3,F1,F2
```

sets the background with the lowest, and foreground-two with the highest priority in a four-partition system.

RESET

The RESET command or statement (Reset I/O Assignments) resets temporary I/O assignments to the standard assignments in the partition in which RESET was submitted. The standard assignments are those specified when the system is generated, plus any modifications made by the operator via an ASSGN command without the TEMP option.

When the physical device affected by RESET is a magnetic tape drive, the current mode set in the PUB table is set to the standard mode set for the device. The standard mode set is established at IPL time and is modified by a permanent ASSGN with an X'ss' parameter.

JCS Format

```
        (SYS    )
// RESET(PROG   )
        (ALL    )
        (SYSxxx )
```

JCC Format

```
     (SYS    )
RESET(PROG   )
     (ALL    )
     (SYSxxx )
```

SYS Resets all system logical units to their standard assignments.

PROG Resets all programmer logical units to their standard assignments.

ALL Resets all logical units to their standard assignments.

SYSxxx Resets the logical unit specified to its standard assignment. SYSIN or SYSOUT cannot be specified.

ROD

The ROD command (Record on Demand) updates all statistical data record (SDR) counters for all non-teleprocessing devices on the recorder file on SYSREC from the SDR counters in storage. The command must not be issued until all jobs in the partitions in an MPS environment have completed. The ROD command also initializes the writing of the RDE end-of-day (EOD) record on SYSREC. The ROD command has no operand.

Note: This command must not be specified for a Model 115 or 125 that is generated without software recording.

JCC Format

ROD

RSTRT

The RSTRT statement (Restart Checkpointed Program) is available for checkpointed programs. A programmer can use the CHKPT macro instruction in his program to write checkpoint records. The maximum number of checkpoints that can be taken is decimal 9999. The checkpointed information includes the registers, tape-positioning information, a dump of the program, and a restart address.

The restart facility allows the programmer to continue execution of an interrupted job at a point other than the beginning. The procedure is to submit a group of job control statements including a restart (RSTRT) statement.

JCS Format

```
// RSTRT SYSxxx,nnnn[,filename]
```

SYSxxx Symbolic unit name of the device on which the checkpoint records are stored. This unit must have been previously assigned.

nnnn Identification of the checkpoint record to be used for restarting. This serial number is four characters. It corresponds to the checkpoint identification used when the checkpoint was taken. The serial number is supplied by the checkpoint routine, and printed on SYSLOG when the checkpoint is taken.

filename Symbolic name of the disk checkpoint file to be used for restarting. It must be identical to the filename of the DTFPH to describe the disk checkpoint file and the fifth parameter of the CHKPT macro instruction. This operand applies only when specifying a disk as the checkpoint file.

See DOS/VS Supervisor and I/O Macros, GC33-5373, for further details on the CHKPT macro instruction.

When a checkpoint is taken, the completed checkpoint is noted on SYSLOG. Restarting can be done from any checkpoint record, not just the last. The jobname specified in the JOB statement must be identical to the jobname used when the checkpoint was taken. The proper I/O device assignments must precede the RSTRT control statement.

Assignment of input/output devices to symbolic unit names may vary from the initial assignment. Assignments are made for restarting jobs in the same manner as assignments are made for normal jobs. Care must be taken that a real-mode program is restarted in a real partition and a virtual-mode program in a virtual partition.

SET

The SET command initializes the UPSI configuration, specifies the number of lines to be printed on SYSLST, specifies the remaining disk capacity when SYSLST or SYSPCH is assigned to disk, and defines to the system the status of the recorder file on SYSREC used by the recovery management support recorder (RMSR) features. It also sets the system date, defines the status of the hard copy file for the display operator console (DOC), defines a new size for the SVA, and specifies an SDL in the SVA. The SET card should precede the JOB card in job control sequence.

Note: RCPCH and RCLST operands are ignored if SYSLST or SYSPCH is assigned to a 3540 diskette.

JCC Format

```
SET [UPSI=n1][,LINECT=n2][,RCLST=n3]
    [,RCPCH=n4][,RF=n5][,DATE=n6][,HC=n7]
    [,SVA=n8][,SDL=n9]
```

UPSI=n1 Sets the bit configuration of the UPSI byte in the communications region. n1 consists of one to eight digits, either 0, 1, or X. Positions containing 0 are set to 0; positions containing 1 are set to 1; positions containing X are unchanged. Unspecified rightmost positions are assumed to be X.

LINECT=n2
Sets the standard number of lines to be printed on each page of SYSLST. n2 is an integer between 30 and 99.

RCLST=n3 n3 is a decimal number indicating the minimum number of records remaining to be written on SYSLST when assigned to disk before a warning is issued to the operator that the capacity of the extent is near. It may be any decimal number from 100 through 65535.

Note: This warning is issued only between jobs. If the extent limits are exceeded before the job, this job is terminated.

If no value is given, the system sets RCLST equal to the value specified in the SYSFIL parameter when the system was generated. If no value was specified, the system sets RCLST equal to 1000.

RCPCH=n4 n4 is a decimal number indicating the minimum number of records remaining to be written on SYSPCH when assigned to disk before a warning is issued to the operator that the capacity of the extent is near. It may be any decimal number from 100 through 65535.

Note: This warning is issued only between jobs. If the extent limits are exceeded before the job, this job is terminated.

If no value is given, the system sets RCPCH equal to the value specified in the SYSFIL parameter when the system was generated. If no value was specified, the system sets RCPCH equal to 1000.

RF=n5 Defines to the system the status of the recorder file (IJSYSRC) on SYSREC used by the recovery management support recorder (RMSR) feature. May only be specified after IPL before the first JOB card.

Note that a SET RF command is ignored if it is given for a Model 115 or 125 without software recording.

n5 can be:

YES Indicates that an active recorder file exists. The system opens this file when the first JOB card is encountered.

CREATE Instructs the system to create a recorder file when the first JOB card is encountered.

DATE=n6 Sets the system date permanently to the specified value. The system date in the communications region of each partition is reset to reflect the new value. This subsequently resets the JOB date when a new job is run. n6 can have the following formats:

mm/dd/yy
dd/mm/yy

mm specifies the month; dd specifies the day; yy specifies the year. The format to be used is the format that was selected when the system was generated.

The DATE parameter may only be specified if the TOD clock is not supported in the system or if the clock is not operational.

HC=n7 Defines to the system the status of the hard-copy file (IJSYSCN) on SYSREC, used to produce hard copy of text that appears on the screen of the Display Operator Console for Model 115 or 125. Each line written on the screen, either by the system or by the operator, is written into IJSYSCN.

n7 can have the following values:

$$\left\{\begin{array}{l}\underline{\text{YES}}\\ \text{NO}\\ \text{CREATE}\end{array}\right\}$$

YES Indicates that a hard-copy file exists in the system and is opened as soon as the first JOB statement is read.

NO Indicates that no recording is to be performed on the hard-copy file.

CREATE Instructs the system to create a hard-copy file; the file is created and opened as soon as the first JOB statement is read.

SVA=n8 Allows the user to change the SVA allocation specified at supervisor generation time. The format of n8 is (nK,nK). The first parameter specifies the size of the SVA (shared virtual area), including the SDL (system directory list), and the system GETVIS area. This size should be an even value and at least 64K. The second parameter specifies the size of the system GETVIS area. This size should be a multiple of 2K and smaller than the size of the SVA. SET SVA may only be specified as the first statement after IPL (or as the first statement of the first procedure).

SDL=n9 The specification of n9 is CREATE, which causes job control to build a system directory list in the SVA. It can also make it possible to load phases into the SVA. SET SDL may only be specified after SET SVA or as the first statement after IPL. SET SDL will not be accepted in a single-partition system.

To build the SDL, job control reads the phasenames that should go into the SDL. These should be specified in subsequent statements each having the following format:

phasename[,SVA]

in which SVA indicates that the phase is to be placed in the shared virtual area if the phase is SVA-eligible. Each entry is 34 bytes long and a maximum of 32K bytes can be entered into the SDL. When this number is exceeded, no more phasenames are accepted and a message is issued to the operator.

The last of the phasenames is indicated when a /* statement is encountered. The SET SDL command, phasenames, and /* can be placed in a procedure. Refer to DOS/VS System Generation, GC33-5377, for details on the IBM SDL procedures distributed with the system.

If the requested phase is not present in the system core image library, a dummy entry is created that is filled when the specific phase is cataloged.

START

The START command (Start or Continue Processing) can be used to start or continue processing in the specified partition. The function of the START command is exactly the same as that of the BATCH command.

AR Format

```
        ┌BG┐
        │F1│
START   │F2│
        │F3│
        └F4┘
```

BG Job control reads the next control statement from SYSLOG.

F1-F4 Specifies that an inactive foreground partition or a foreground partition that has been stopped by a STOP command is to be restarted. See also the BATCH command.

STOP

The STOP command (Stop Processing) indicates that there are no more jobs to be executed in the partition in which the command is given. It cannot be used in a single-partition system.

JCC Format

```
STOP
```

This command removes the partition from the system's task selection mechanism. Job control remains in the partition and can be activated by the START or BATCH attention routine command.

Notes:

1. It may sometimes be advisable to use a STOP command instead of a PAUSE command. The PAUSE command issues a read to SYSLOG, tying it up until the operator responds.
2. The STOP command can be used instead of the combination of HOLD and UNBATCH commands.

TLBL

The TLBL (Tape Label Information) statement replaces the VOL and TPLAB statement combination. It contains file label information for tape label checking and writing. (Programming support for the previous VOL and TPLAB combinations continues.) The TLBL statement may be used with both EBCDIC and ASCII files. For ASCII file processing, the fourth and fifth operands are called set identifier and file section number, respectively. For detailed information about TLBL refer to DOS/VS Tape Labels, GC33-5374.

JCS Format for EBCDIC Files

```
// TLBL  filename,['file-id'],[date],
         [file-serial-number],
         [volume-sequence-number],
         [file-sequence-number],
         [generation-number],
         [version-number]
```

JCS Format for ASCII Files

```
// TLBL  filename,['file-id'],[date],
         [set-identifier],
         [file-section-number],
         [file-sequence-number],
         [generation-number],
         [version-number]
```

filename
This can be from one to seven alphameric characters, the first of which must be alphabetic. This unique filename is identical to the symbolic name of the program DTF that identifies the file.

'file-ID'
One to seventeen alphameric characters, contained within apostrophes, indicating the unique name associated with the file on the volume. This operand may contain embedded blanks. On output files, if this operand is omitted, "filename" is used. On input files, if the operand is omitted, no checking will be done.

date Output Files: A retention period in the format d through dddd (0-9999) or yy/ddd (yy=0-99, ddd=1-365) can be specified. If omitted, a 0 retention period is assumed. The current system date is always used as the creation date for output files.

Input Files: A four to six numeric character creation date in the format yy/ddd (99/365) can be specified (ddd can be from 1-365).

If omitted or a retention date is specified, no checking is done for input files.

file serial number (EBCDIC) or
set identifier (ASCII)

One to six alphameric characters indicating the volume serial number of the first (or only) reel of the file. All six characters must be specified for ASCII files. For EBCDIC files, if fewer than six characters are specified, the field is right justified and padded with zeros. If this operand is omitted on output, the volume serial number of the first (or only) reel of the file is used. If the operand is omitted on input, no checking is done.

volume sequence number (EBCDIC) or
file section number (ASCII)

One to four numeric characters in ascending order for each volume of a multiple volume file. This number is incremented automatically by OPEN/CLOSE routines as required. If this operand is omitted on output, BCD 0001 is used. If omitted on input, no checking is done.

file sequence number

One to four numeric characters in ascending order for each file of multiple file volume. This number is incremented automatically by OPEN/CLOSE routines as required. If omitted on output, BCD 0001 is used. If omitted on input, no checking is done.

generation number

One to four numeric characters that modify the file ID. If omitted on output, BCD 0001 is used. If omitted on input, no checking is done.

version number

One or two numeric characters that modify the generation number. If omitted on output, BCD 01 is used for EBCDIC files, and BCD 00 for ASCII files. If omitted on input, no checking is done.

TPBAL

The TPBAL (Teleprocessing Balancing) command allows the operator to display the status of the TP Balancing function, or to change that status.

AR Format

```
TPBAL
```

This format of the TPBAL command displays, on the device assigned to SYSLOG, the present status of the TP Balancing function. It indicates what partitions are currently being affected by the function.

```
TPBAL  n
```

This format of the command alters the status of the TP Balancing function. The operand specifies the number of partitions in which processing can be delayed (always those partitions that have the lowest priorities). The system responds by displaying the partitions that will be affected under the new status. A specification of zero renders the function inoperative. The maximum number that can be specified is the number of partitions in the system minus one.

TPLAB

The TPLAB (Tape Label Information) statement, which can be used for both EBCDIC and ASCII files, contains file label information for tape label checking and riting. It must immediately follow a volume (VOL) statement. The TPLAB statement contains an image of a portion of the standard tape file label. Label fields 3-10 are always included just as they appear in the label. These are the only fields used for label checking. The additional fields (3-13) can be included, if desired. Field 12 (block count) always defaults to zeros. If specified for an output file, they are written in the corresponding fields of the output label. They are ignored when used for an input file. DOS/VS Tape Labels, GC33-5374, contains additional information about TPLAB.

JCS Format

```
// TPLAB {'label-fields 3-10'}
         {'label fields 3-13'}
```

'label fields 3-10'

This is a 49-byte character string, included within apostrophes (8-5 punch), identical to positions 5-53 of the tape file label. These fields can be included in one line.

'label fields 3-13'

This is a 69-byte character string, included within apostrophes (8-5 punch), identical to positions 5-73 of the tape file label. These fields are too long to be included on a single line. The character string must extend into column 71, a continuation character (any character) is present in column 72, and the character string is completed on the next line. The continuation line starts in column 16.

UCS

The UCS command (Load Universal Character Set Buffer) causes the 240-character Universal Character Set contained in the core image library phase specified by phasename to be loaded as buffer storage in the IBM 2821 Control Unit. The 240 EBCDIC characters correspond to the 240 print positions on 1403 chains and trains. A character sent to the printer for printing is matched against the characters in the UCS buffer. When a match occurs, the corresponding chain/train character is printed in the print-line position that the output character occupied. Thus, through the UCS buffer and the many chains/trains available, the 1403 Printer can be adapted to many variable printing applications.

The logical unit must be assigned to an IBM 1403 Printer with the UCS feature. It is the user's responsibility to assemble, link-edit, and catalog his UCS buffer phases into the core image library, and to mount the new chain or train before the UCS command is executed. The UCS command is not logged on SYSLST.

JCC Format

```
UCS SYSxxx,phasename[,FOLD][,BLOCK]
    [,NULMSG]
```

SYSxxx The name of the logical unit assigned to a 1403 UCS printer to be loaded.

phasename

The symbolic name of the core image library phase containing the 240 EBCDIC characters to be loaded, followed by an 80-character verification message. Each phase may have any valid phasename.

FOLD Signifies that the buffer is to be loaded with the folding operation code in the CCW to permit printing either uppercase or lowercase bit configurations.

BLOCK Signifies that the 2821 latch is to be set to inhibit data checks generated by the 1403 UCS Printer because of print line character mismatches with the UCS buffer.

NULMSG Signifies that the 80-character verification message is not to be printed on the 1403 after the buffer is loaded. If this parameter is not specified after the UCS buffer has been loaded, the program skips to channel 1, issues a print of the last 80 characters in the phase specified by the first parameter, and again skips to channel 1. This is to identify the phase, if the phasename is incorporated in the verification message. If a chain/train can be identified by a unique character, this message can also be used to verify that the mounted chain or train is compatible with the UCS buffer contents, by including this unique character in the verification message.

The UCS phase format is:

240-character UCS buffer load	80-character verification message

UNBATCH

The UNBATCH command terminates foreground processing and releases the partition. UNBATCH is accepted only when no job is in process in the partition and only from SYSLOG. The operator can gain command of SYSLOG following a PAUSE or STOP command or a PAUSE statement. All tape or disk system I/O files must have been closed. Following the UNBATCH command, the attention routine accepts BATCH or START commands for the affected partition. UNBATCH permits storage allocation for the partition to be reduced. This command is valid only for foreground partitions.

JCC Format

```
UNBATCH
```

UPSI

The UPSI (User Program Switch Indicators) statement allows you to set program switches that can be tested. It has the following format.

JCS Format

```
// UPSI nnnnnnnn
```

The operand consists of one to eight characters of 0, 1, or X. Positions containing 0 are set to 0. Positions containing 1 are set to 1. Positions containing X are unchanged. Unspecified rightmost positions are assumed to be X.

Job control clears the UPSI byte to zeros before reading control statements for each job. When job control reads the UPSI statement, it sets or ignores the bits of the UPSI byte in the communication region. Left to right in the UPSI statement, the digits correspond to bits 0 through 7 in the UPSI byte. Any combination of the eight bits can be tested by problem programs at execution time.

VOL

The VOL statement (Volume Information) is used when standard labels for a DASD or tape file are checked unless the DLBL or TLBL statements are used. A VOL statement must be used for each file on a multifile volume (when the DLAB or TPLAB statements are used). The VOL, TPLAB or VOL, DLAB, XTENT statements must appear in that order and must immediately precede the EXEC command or statement to which they apply.

The VOL statement, in combination with the DLAB and XTENT statements, should not be used to supply label and extent information for the 3330/3333 or 3340.

JCS Format

```
// VOL SYSxxx,filename
```

SYSxxx
: Symbolic unit name. The symbolic unit name is taken from the XTENT statement.

filename
: This can be from one to seven alphameric characters, the first of which must be alphabetic. This unique filename is identical to the symbolic name of the program DTF that identifies the file.

XTENT

The XTENT statement (DASD Extent Information) defines each area, or extent, of a DASD file. One or more XTENT statements must follow each DLAB statement.

The XTENT statement, in combination with the DLAB and VOL statement, should not be used to supply label and extent information for the 3330/3333 or 3340. The XTENT statement cannot be used for VSAM files.

JCS Format

```
// XTENT type,sequence,lower,upper,
         'serial no.',SYSxxx[B2]
```

type
: Extent Type. One or three columns, containing:

1 = data area (no split cylinder)

2 = overflow area (for indexed sequential file)

4 = index area (for indexed sequential file)

128 = data area (split cylinder). If type 128 is specified, the lower head is assumed to be $H_1H_2H_2$ in lower, and the upper head $H_1H_2H_2$ in upper.

sequence
: Extent Sequence Number. One to three columns, containing a decimal number from 0 to 255, indicating the sequence number of this extent within a multiextent file. Extent sequence 0 is used for the master index of an indexed sequential file. If the master index is not used, the first extent of an indexed sequential file has sequence number 1. The extent sequence for all other types of files begins with 0.

lower
: Lower Limit of Extent. Nine columns, containing the lowest address of the extent in the form $B_1C_1C_1C_2C_2C_2H_1H_2H_2$, where:

B_1 = initially assigned cell number.

0 for disk
0 to 9 for data cell

C_1C_1 = subcell number.

00 for disk
00 to 19 for data cell

$C_2C_2C_2$ = cylinder number.

000 to 199 for disk

or

strip number:

000 to 009 for data cell

H_1 = head block position.

0 for disk
0 to 4 for data cell

H_2H_2 = head number

00 to 09 for 2311
00 to 19 for 2321, 2314, 2319, 3330/3333, 3340.

Although a part of the address (such as B_1 or $C_2C_2C_2$) can be zero, a lower extent of <u>all</u> zeros is invalid.

<u>Note</u>: The last four strips of subcell 19 are reserved for alternate tracks for 2321.

upper Upper Limit of Extent. Nine columns containing the highest address of the extent, in the same form as the lower limit.

'serial no.' Volume Serial Number. This is a six-byte alphameric character string, contained within apostrophes. The number is the same as in the volume label (volume serial number) and the Format 1 label (file serial number).

SYSxxx This is the symbolic address of the DASD drive.

B_2 Currently assigned cell number.

0 for disk
0-9 for data cell

This field is optional. If missing, job control assigns $B_1=B_2$.

ZONE

The ZONE statement initializes the value of the job zone field in the communication region (bytes 143 and 144). If no ZONE statement is supplied, job control supplies the zone given in the system zone field in the communication region extension (bytes 68 and 69). If no DATE statement is supplied, the job date is updated by means of the values given in the system date field and in the ZONE statement.

Locations that are on Greenwich Mean Time need not specify the ZONE statement or can specify

```
// ZONE EAST/00/00 or
// ZONE WEST/00/00
```

<u>JCS Format</u>

```
// ZONE{EAST} /hh/mm
       {WEST}
```

EAST A geographical position east of Greenwich.

WEST A geographical position west of Greenwich.

hh/mm A decimal value that indicates the difference in hours and minutes between local time and Greenwich Mean Time. hh may be in the range 0-12, mm in the range 0-59.

This statement is only accepted if time-of-day clock support is included in the system. Otherwise, the message 1S0nD INVALID STATEMENT appears on SYSLOG and SYSLST.

/+ -- End-of-Procedure

The End-of-Procedure statement is /+, unless you specify a different end-of-procedure statement as indicated in the EOP parameter of the CATALP statement. This statement is valid only for cataloged procedures and must be the last statement of each procedure to be cataloged. (For information on the EOP parameter, refer to the CATALP statement in the section Procedure Library: Maintenance and Service Programs.)

<u>JCS Format</u>

```
/+ [comment]
```

Columns 1 and 2 contain a slash (/) and a plus (+). Column 3 must be blank.

As a special delimiter statement, /+ is neither logged nor listed when retrieving a cataloged procedure for inclusion in the job input stream. Instead, the following message appears:

```
EOP procedurename
```

The /+ statement is punched and/or listed only when a procedure is processed by the PSERV program.

/* -- End-of-Data File

The End-of-Data File statement must be the last statement of each input data file on SYSRDR and SYSIPT.

JCS Format

/*

Columns 1 and 2 contain a slash (/) and an asterisk (*). Column 3 must be blank. /* causes the channel scheduler to post the end-of-file indicator in the user's CCB. Logical IOCS also recognizes /* when a card reader is assigned to the symbolic units SYS000-SYSnnn.

Note: For an input file on an IBM 5425 MFCU, the /* card must be followed by a blank card.

/& -- End-of-Job

This End-of-Job statement must be the last statement of each job.

JCS Format

/& [comments]

Columns 1 and 2 contain a slash (/) and an ampersand (&) (12-punch). Column 3 must be blank. Upon occurrence of /&, the channel scheduler posts an end-of-file indicator in the user's CCB. If the user attempts to read past the /& on SYSRDR or SYSIPT, the job is terminated. Any comments can begin in column 14 and are printed at end of job. If a job updates a system directory, comments included on the /& statement are not printed.

If time-of-day clock support is provided, the end-of-job statement is printed on SYSLST in the following format: columns 1-3 contain EOJ, columns 5-12 the job name columns 14-72 blanks or any user comments, and columns 73-118 the date, time-of-day, and job duration in the following format:

DATE mm/dd/yy,CLOCK hh/mm/ss,
DURATION hh/mm/ss

On SYSLOG, the date, time of day, and job duration (the amount of time elapsed between the start and the end of a job) appear in the same format, occupying 46 positions, on the line following the end-of-job statement.

If time-of-day clock support is part of your system, the zone and date values are reset every time this statement is encountered.

Note: The stop time that the job accounting routines stores in byte 40 of the Job Accounting Table is calculated from the values in storage locations 80 and 84.

End-of-job information is not printed on SYSLST if // OPTION NOLOG has been specified. (The NOLOG statement itself is logged on SYSLST).

* -- Comments

This statement can be used as a job control comments statement.

JCS Format

* any user comments

Column 1 contains an asterisk. Column 2 is blank. The remainder of the statement (through column 72) contains any user comments. The content of the comment statement is printed on SYSLOG. If followed by a PAUSE statement, the statement can be used to request operator action.

JOB CONTROL STATEMENT EXAMPLE

Figure 6 contains six examples of job control statement input. In the discussion that follows, each point corresponds to the number at the left of the two slashes in the job control statements. The PHASE, INCLUDE, and ENTRY statements are linkage editor control statements. These statements are described in detail in the section Linkage Editor. They are included in this discussion to present a more meaningful example.

1. JOB statement for first example.

2. ASSGN statement for the disk with volume serial number 231402.

3. DLBL and EXTENT statements to define ISAM file to be created.

4. EXEC statement for a program in the system core image library to be executed in virtual mode.

5. End-of-job indicator. All temporary symbolic unit assignments reset to the permanent assigns.

6. JOB statement for example 2.

7. ASSGN statements for 3330 disks and tape.

8. DLBL and EXTENT statements to define a sequential disk file with two extents on separate 3330 volumes.

9. EXEC statement for a program in the system core image library that is to be executed in REAL mode, using 60K of the real storage allocated to BGR.

10. MTC (Magnetic Tape Command) to rewind and unload the tape just created.

11. Message to operator concerning the tape.

12. End-of-job indicator.

13. MAP command to print a map of real and virtual storage allocations on SYSLOG.

14. JOB statement for example 3.

15. ASSGN statements to release previous tape assignment, and temporarily assign SYSPCH to that tape.

16. OPTION statement to specify options that are different from those established at System Generation.

17. CATALR statement that will be transferred by Job Control to the SYSPCH file (tape on 381). This tape can then (after creation of the object deck) be used as input to the MAINT program to catalog to the relocatable library.

18. EXEC statement for the system assembler.

19. Source deck as input to the system assembler and /* (end-of-data).

20. MTC commands to write a tapemark and rewind the tape on 381. This tape is now positioned for use as SYSIPT.

21. RESET statement causes SYSPCH to revert to its permanent assignment.

22. Temporary assignment of SYSIPT to tape 381 for use by the librarian program MAINT.

23. EXEC statement for MAINT.

24. MTC command to rewind and unload tape on unit 381.

25. End-of-job indicator with a comment. SYSIPT returns to its permanent assignment.

26. JOB statement for example 4.

27. Permanent assignment of private core image library to 3330 on unit 110.

28. OPTION statement to specify the CATAL function is to take place.

29. PHASE and INCLUDE statements that are input to the linkage editor. The first INCLUDE statement calls the module previously cataloged in the relocatable library and the second (with a blank operand) is followed by an object deck to be included.

30. /* indicates the end of the object deck, not the end of input to the linkage editor.

31. ENTRY statement input to the linkage editor specifying an entry point for the PHASE PROGX03.

32. EXEC statement for the linkage editor.

33. Permanently unassigns the private core image library.

34. End-of-job indicator.

35. JOB statement for example 5.

36. EXEC statement using cataloged procedures. LIBMAINT is a set of JCL and SYSIPT input in the Procedure Library.

37. End-of-job indicator.

38. JOB statement for example 6.

39. ASSGN statement for SYSLST, which may be any printer.

40. SYS004 should be assigned to a tape on X'380', or X'381' (if X'380' is not available), or X'382' (if X'381' is not available).

41. Assign SYS005 to the same unit as SYS004 (described in 40).

```
1    // JOB B79ISF                   LOAD ISAM FILE
2    // ASSGN SYS005,DISK,VOL=231402  ISAM BACKUP MASTER -- PREMOUNTED PACK
     // DLBL LOADIS,'FILE8685',,ISC
     // EXTENT SYS005,231402,4,1,200,20
3    // EXTENT SYS005,231402,1,2,300,360
4    // EXEC ISOOF
5    /&

6    // JOB U81SDC                 UNLOAD SEQUENTIAL DISK TO TAPE
     // ASSGN SYS004,X'111'        INPUT MASTER FILE
7    // ASSGN SYS005,X'112'        (2 EXTENTS)
     // ASSGN SYS006,X'380',X'C8'  BACKUP TAPE DUAL DENSITY 9-TRACK
     // DLBL SDUNLD,'SEQUENTIAL FILE',73/206,SD
8    // EXTENT SYS004,333002,1,0,1900,380
     // EXTENT SYS005,333003,1,1,76,570
9    // EXEC SD008,REAL,SIZE=60K   RUN IN REAL USING 60K
10   // MTC RUN,SYS006
11   * OPERATOR - TAPE ON 380 - LABEL, REMOVE RING
     *                  AND RETURN TO LIBRARY
12   /&
     * CREATE A MAP OF STORAGE ON SYSLOG
13   MAP

14   // JOB R61ASSM                OBJECT DECK TO TAPE - CATALOG IN
     *                             RELOCATABLE LIBRARY
     ASSGN SYS012,UA               CLEAR PREVIOUS TAPE ASSIGN
15   // ASSGN SYSPCH,X'381'        ASSIGN SYSPCH TO TAPE
16   // OPTION DECK,LIST,XREF,NOEDECK
17    CATALR MOD207
18   // EXEC ASSEMBLY
19    (ASSEMBLER SOURCE HERE)
     /*
20   // MTC WTM,SYSPCH,02          WRITE TAPEMARK AND
     // MTC REW,SYSPCH             REWIND SYSPCH TAPE
21   // RESET SYSPCH
22   ASSGN SYSIPT,X'381',TEMP TO READ MODULE PRODUCED ABOVE -- TEMP ASSGN OVERRIDE
23   // EXEC MAINT
24   MTC RUN,X'381' UNLOAD/REWIND SYSIPT
25   /&                            EOJ R61ASSM

26   // JOB K13CATL                LINK MODULES INTO A
27   ASSGN SYSCLB,X'110'           PRIVATE CORE IMAGE LIBRARY
28   // OPTION CATAL
        PHASE PROGX03,*
29    INCLUDE MOD207
      INCLUDE
      (OBJECT DECK INCLUDED HERE)
30   /*
31    ENTRY MD207B
32   // EXEC LNKEDT
33   ASSGN SYSCLB,UA
34   /&

35   // JOB E39MANT                CATALOGED PROCEDURE FOR LIBRARY MAINTENANCE
36   // EXEC PROC=LIBMAINT
37   /&
38   // JOB E40
39   // ASSGN SYSLST,PRINTER       ASSIGN SYSLST TO ANY PRINTER
40   // ASSGN SYS004,(X'380',X'381',X'382') ASSIGN SYS004 TO A TAPE WITHIN THIS RANGE
41   //ASSIGN SYS005,SYS004 ASSIGN SYS005 AS SYS004
     // EXEC MYPROG
     /&
```

Figure 6. Job Control Statement Examples

POWER/VS

POWER/VS is a reader/writer spooling processor that provides automatic staging of unit-record input and output, class scheduling, and priority scheduling for all DOS/VS programs executed under its control. The remote job entry (RJE) facility of POWER/VS permits jobs to be entered into the central system from up to 25 terminals concurrently.

This section describes the POWER/VS job entry control language, the central operator commands, and the POWER/VS RJE terminal commands.

A summary of all POWER/VS statements and commands is contained in Appendix C.

Job Entry Control Language (JECL)

The job entry control language (JECL) specifies how POWER/VS is to handle a job. If JECL is used (it is not required), it supplements the DOS/VS job control language by allowing the user more options.

Since JECL statements begin with an asterisk-blank, job control treats them as comments when POWER/VS is not in operation.

This also means that for a writer-only system, dual entries are created for LST, PUN, and PRT statements. To prevent this, "OPTION NOLOG" should precede the JECL statement.

Functions of JECL Statements

JECL allows you to specify how POWER/VS is to handle a particular job. You may use JECL statements, for instance, for the following:

1 Under POWER/VS and POWER/VS RJE:

- Assign priorities to determine the order in which the jobs are started.
- Assign input and output dispositions. (Determine the way in which POWER/VS routes and schedules queue entries.)
- Spool output to tape or disk intermediate storage.
- Specify the partition in which the job is to be executed.
- Segment the output. (Break bulky list or punch output into segments so that printing or punching can be started before execution of the generated user program has been completed.)

2 Under POWER/VS RJE only:

- Direct output to another user or to all terminal users.
- Direct output to the central installation.

Format of JECL Statements

JECL statements are entered as punched cards within the job stream. The format of JECL statements is shown in Figure 7.

Columns 1-4	5-71			72	73-80
* $$ ①	Operation Field ②	Operand Field ③	Comments Field ④		Sequence Field ⑤

Figure 7. Format of JECL Statements

Each JECL statement consists of following fields:

1. Identification field: Contains the characters * $$ in columns 1 through 4. (Column 2 must be blank.) Can be used for POWER/VS purposes only.
2. Operation field: Specifies the JECL operation. It may either directly follow the second dollar sign or be separated from the second dollar sign by one or more blanks. At least one blank must separate the operation field from the operand field.

3. Operand field: Contains one or more positional or keyword parameters, separated by commas. A blank terminates the operand field; therefore, no parameter may contain embedded blanks. Keyword or positional parameters cannot be mixed within one statement.

   Note: Some POWER/VS features are supported only through keyword notations.

   Positional parameters must be specified in a fixed order. If a positional parameter is omitted, its delimiting comma must be coded. If trailing parameters are omitted, trailing commas can also be omitted. If no parameters are specified, the entire operand field can be omitted. When a comment field is present and the entire operand field is blank, the omitted operand field is indicated by a comma, which is preceded and followed by at least one blank. If the operand field cannot be contained on one card, a continuation card can be produced according to the following rules:

   - The operand field can be broken after a comma that separates two parameters, and can be continued on the next card starting from columns 6 through 16. Column 72 of the first card must contain a non-blank character. This process can be repeated for as many cards as are necessary.

   - Columns 1 to 4 of each continuation statement must contain * $$ (asterisk-blank-dollar-dollar). Column 72 of a one-card statement or a last continuation card must be blank.

   Note: In a writer-only partition, continuation of a * $$ JCB card is not recognized.

4. Comments field: May contain any information. This field can be broken off anywhere and continued on the next card in any column, subject to the same rules given for the operand field.

5. Sequence field: May contain up to eight characters of optional information used for control statement identification. If present, the sequence field is positionally dependent and must start in column 73.

Description of JECL Statements

Figure 8 lists all JECL statements in alphabetical order. This list is followed by a detailed description of each statement and its operands. Examples demonstrating the use of JECL statements are given at the end of this section.

JECL Statement	Function
* $$ CTL	Specifies a default input class.
* $$ DATA	Inserts data into a book retrieved from a source statement library.
* $$ EOJ	Indicates the end of a POWER/VS job.
* $$ JOB	Indicates the beginning of a POWER/VS job and provides handling information.
* $$ LST * $$ PRT	Provides handling information for printed output.
* $$ PUN	Provides handling information for punched output.
* $$ RDR	Inserts a diskette file into the input stream.
* $$ SLI	Inserts data from a source sublibrary into the job stream.
* $$/*	Used in a source statement library book to indicate the end of a DOS/VS job step (used with the SLI statement only).
* $$/&	Used in a source statement library book to indicate the end of a DOS/VS job (used with the SLI statement only).

Figure 8. POWER/VS JECL Statements

The following pages describe all the the POWER/VS JECL statements in alphabetical order. A summary of these statements is contained in Appendix C.

* $$ CTL Statement

The * $$ CTL statement establishes the default input class that is assigned to all jobs whose input class is not specified in the * $$ JOB statement. It overrides an input class specification that may have been made in a PSTART command. It may be placed at any POWER/VS job boundary in the input stream, and becomes effective immediately. It remains in effect until the next * $$ CTL statement is read or until the reader task is stopped with a PSTOP command.

Keyword Format only

```
* $$ CTL  ┌CLASS=┌A    ┐┐
          └      └class┘┘
```

class Specifies the input class to which any subsequent jobs are assigned if they do not contain a class specification in their * $$ JOB statement. The CLASS parameter may be used in order to group jobs that require the same I/O configuration, partition, or other resources for execution. If class is not specified, class A is assumed. Partition-dependent; class is specified as a single alphabetic character (A-Z) or a single number (0 through 4). When specified as a number, input class is 0 through 4.

The CLASS parameter in the * $$ CTL statement overrides the class specified in the PSTART command, which overrides default class A.

* $$ DATA Statement

The * $$ DATA statement allows you to insert a deck of cards from the card reader into a book that is retrieved from the source statement library by an * $$ SLI statement. Data statements must be preceded by an * $$ SLI statement; otherwise, they are treated as comments. * $$ SL update statements (* $$ SLI in columns 73-76) and * $$ DATA statements may be used together. However, they must appear in the job stream in the same order as the corresponding images appear in the source statement library book. The statement is ignored in a writer-only partition.

Format

```
* $$ DATA   name
```

name Specifies the name of the corresponding * $$ DATA statement in the source statement library book where data is to be inserted. name can be from one to eight alphameric characters. The first character must be alphabetic.

* $$ EOJ Statement

The * $$ EOJ statement marks the end of a POWER/VS job and, if used, is normally placed at the end of a DOS/VS job or job step. If this statement is submitted at any other time than DOS/VS job control time, it is still acted upon by POWER/VS.

Format

```
* $$ EOJ
```

* $$ JOB Statement

The * $$ JOB statement may be placed anywhere in a DOS/VS job stream to specify POWER/VS job attributes or to begin a POWER/VS job. An * $$ JOB statement encountered in the input stream without a preceding * $$ EOJ statement is used as a delimiter for the previous POWER/VS job. If the * $$ JOB statement is omitted, POWER/VS jobs are delimited by the // JOB and /& job control statements. The POWER/VS job name is then copied from the DOS/VS job name.

Positional Format

```
* $$ JOB       ┌AUTONAME┐
               └jobname ┘
              ,┌D          ┐
               └disposition┘
              ,[priority]
              ,[class]
```

Keyword Format

```
* $$ JOB      ┌     ┌AUTONAME┐┐
              └JNM= └jobname ┘┘
              ┌      ┌D          ┐┐
              └,DISP=└disposition┘┘
              [,PRI=priority]
              [,CLASS=class]
              [,USER=user-information]
```

jobname Specifies the name by which the job and its queue entries are known to POWER/VS. If the job contains several DOS/VS jobs, they are all logged in POWER/VS by the jobname in the * $$ JOB statement. The default name AUTONAME is assigned if jobname is not specified. jobname may be specified as one to eight alphameric characters. The first character must be alphabetic. The use of the jobname ALL, HOLD, FREE, RJE, LOCAL, or one-character jobnames should be avoided as they may conflict with some POWER/VS operator commands.

disposition
Specifies how POWER/VS will route and schedule the associated entry in the read queue. It may be one of the following: H for hold, K for keep, L for leave, or D for delete after processing. The default is D.

D - Delete after processing. The job is automatically scheduled by POWER/VS according to its class and priority. After job execution, the read queue entry is deleted from the read queue and the data file space is released.

H - Hold job. The job remains in the read queue; it is not dispatched by POWER/VS until the operator changes the disposition to D or K by means of the PALTER command or issues the PRELEASE command.

K - Keep after processing. The job will be automatically scheduled by POWER/VS according to its class and priority. After job execution, the read queue entry is not deleted from the read queue, but the disposition becomes L.

L - Leave in queue. The job remains in the read queue; it is not dispatched by POWER/VS until the operator changes the disposition to K or D by means of the PALTER command or issues the PRELEASE command.

Disposition is ignored in a writer-only partition.

priority Specifies the priority assigned to this job in both input and output queues. It is specified as a single digit from 0 to 9. Nine is the highest priority. If the priority parameter is not specified, POWER/VS assigns the default priority defined in the PRI parameter of the POWER macro. Jobs are dispatched on a first-in-first-out basis within priority, within class. Priority is ignored in a writer-only partition.

class Specifies the class that is assigned to the job entry in the input queue. It may be specified as an alphabetic character from A through Z, or as a numeric character from 0 through 4. This parameter may be used to group jobs that require the same I/O configuration, partition, or other resources for execution. Class is ignored in a writer-only partition.

Classes 0 through 4 have a specific meaning and are called the partition-dependent classes. Jobs with class 0 can only be selected for execution in the background partition, jobs with class 1 in partition F1, and so on.

If the class parameter is omitted, the class specification is taken from the CTL statement. If no CTL statement is specified, the class is established by the PSTART command. If no class is specified at all, default class A is assigned.

Note: When the * $$ JOB statement is coded in its positional form, class 0 may be coded as BG, and classes 1 through 4 may be coded as F1 through F4.

user information
Up to 16 bytes of user information may be specified. If JLOG=YES was specified during during POWER/VS generation, this information is printed on SYSLOG when the execution read task encounters an * $$ JOB statement. The POWER/VS user information also appears in the POWER/VS account records and in separator pages. User information must be specified as alphameric unless within quotes.

* $$ LST or PRT Statement

The * $$ LST (or * $$ PRT) statement defines the attributes of the intercepted list output of a POWER/VS job. LST and PRT are equivalent operation codes. Any number of * $$ LST statements is permitted per POWER/VS job. If the * $$ LST statement is not included in the job, the default values established during POWER/VS generation are assumed. * $$ LST statements do not require that * $$ JOB and * $$ EOJ statements also be used.

The specifications of a particular * $$ LST statement remain in effect until a subsequent * $$ LST statement with the same parameter is encountered, or until the end of the POWER/VS job.

For each * $$ LST statement within a POWER/VS job, a separate list queue entry is created (data-driven output segmentation).

Positional Format

```
* $${LST} [D              [A    ]]
    {PRT} [disposition    [class]]
         , [ƀƀƀƀ        ]
           [forms-number]
           [1               ]
         , [number-of-copies]
           [tape-devaddr    ]
         , [norbm1]
         , [linetab]
```

Keyword Format

```
* $${LST} [DISP={D          }]
    {PRT} [      {disposition}]
          [,CLASS={A    }]
          [       {class}]
          [,REMOTE=remid]
          [,FNO={ƀƀƀƀ        }]
          [     {forms-number}]
          [,JSEP=sep]
          [,COPY={1               }]
          [      {number-of-copies}]
          [,TADDR=tape-devaddr]
          [,RBM=(norbm1,norbm2)
          [,LTAB=linetab]
          [,RBS=norbs]
          [,UCS=(phasename[,option])]
          [,FCB=phasename]
          [,LST=listaddr]
```

disposition
: Specifies the disposition of the printed output after execution of the job. The disposition specification options are listed below:

D - Delete after processing. The
or list queue entry is written to
R a printer according to its class and priority.

H - Hold. The list queue entry remains in the list queue; it is not written to a printer by POWER/VS until the operator changes the disposition to D or K by means of the PALTER command or issues the PRELEASE command.

K - Keep after processing. The list queue entry is automatically written to a printer according to its class and priority. When printing is completed, the list queue entry is not deleted from the list queue, and the disposition of the queue entry becomes L.

L - Leave in queue. The list queue entry remains in the list queue; it is not written to a printer until the operator changes the disposition to D or K by means of the PALTER command or issues the PRELEASE command.

N - Output without spooling. If N is specified, print requests are not intercepted. Output is written directly, without spooling. If the N option is specified, and the device to which SYSLST is assigned is not available at execution time, message 1Q46I is issued and punch output is spooled.

T - Spool to tape intermediate storage. PSTART is used to initiate printing.

If the disposition specification is omitted, D is assumed.

D, H, K, and L are only valid when the output is spooled to disk.

When the output is to be printed at a terminal, N and T are invalid options.

class
: Specifies the class of the printed output after execution of the job. It may be specified as any alphabetic character (A through Z). Class may be used in conjunction with forms-number to group together the same types of printed output. Class is only meaningful when the output is spooled to disk. If class is specified and the positional form is used, disposition must also be specified. If class is omitted, A is assigned. If the * $$ LST statement is omitted, output class can be specified in the PSTART command.

Note: There is no comma between disposition and class in the positional form.

remid
: Specifies the TO remote-id to which list output is to be routed. Each list queue entry also has assigned to it a FROM remote-id, which identifies the user who initially submitted the job. remid can be a number from 0 to 200. Numbers 1 to 200 represent specific remote users. If remid is 0, list output is directed to the central installation. If this parameter is omitted or if the positional form of the * $$ LST statement (which does not support

this parameters) is used, list output is routed to the central location if the job was submitted locally, or is routed according to the LSTROUT specification in the PRMT macro if the job was submitted remotely.

forms-number
: Specifies the type of forms to be used according to installation requirements. Forms-number is specified by one to four alphameric characters. If forms-number is not specified, it is assumed to be four blanks, representing the installation's standard type of forms. When a list writer task is to process output from a list queue entry, it determines whether this forms-number matches that of the previous list queue entry. If they match, the list writer task processes the output. If they do not, message 1Q40A is issued. If the first queue entry processed after a PSTART has a forms ID of four blanks, no mount message is issued.

sep
: Specifies the number of separator pages (0 to 9). If not specified the value defined during POWER/VS generation in the first entry of the JSEP parameter of the POWER macro is assumed.

Each separator page contains the POWER/VS job name and number, user information from the * $$ JOB card, and the date and time. Separator pages that delimit the last or only segment contain also the word "LAST".

number-of-copies
: Specifies the number of copies of printed output from the list queue entry. This parameter is ignored when the output is not spooled to disk. The parameter is specified by one or two digits. If omitted or 0 or 00 is specified, one copy is printed. If additional copies are required when disposition is T, the tape must be reprinted.

tape-devaddr
: Specifies the hardware address of the tape drive to be used for spooling to tape intermediate storage when the job is being executed. This parameter can be specified as one of the following:

```
cuu
X'cuu'
(cuu,X'ss')
(X'cuu',X'ss')
(cuu,ss)
(X'cuu',ss)
```

X'ss' specifies the density of the output tape. tape-devaddr forces a disposition of T. If DISP=T is specified and tape-devaddr is omitted, the operator is requested to specify the tape address at job execution time.

Multiple files can be written to a single tape. Multivolume files are also supported. No label checking is performed.

norbml
: Specifies the number of pages (skips to channel1) intercepted before message 1Q52I is sent to the operator. This parameter is specified by one to six digits. If norbml is omitted, the value defined during POWER/VS generation in the first entry of the STDLINE parameter of the POWER macro is assumed.

norbm2
: Specifies the additional number of list records intercepted each time before message 1Q52I is issued again. It is specified by one to six digits. If norbm2 is omitted, the preceding comma as well as the parentheses may be omitted and the value defined during POWER/VS generation in the second entry of the STDLINE parameter of the POWER macro is assumed. This value is also assumed when the positional form of the LST statement is used.

linetab
: Specifies the carriage control tape format to be used wheh print requests are intercepted. This parameter is specified as 26 digits. If linetab is not specified, the value of the LTAB parameter specified in the POWER macro during POWER/VS generation is used. linetab specifies the distance between the first carriage control tape channel punch and the other channel punches and the end of the page. It is specified as d0d1...d11d12, where each d represents two digits, and where d0 is the distance between the channel 12 and the first punch; d1 is the distance between the first punch and the channel 1 punch; d2 is the distance between the first punch

and the channel 2 punch, and so forth. If channel 1 is the first punch, distance d1 is expressed as 00. If a channel is not punched, 00 must be specified for that channel.

The sum of d0 and d12 is assumed to specify the maximum number of lines on a page. If d12 is specified as zero, a default value is taken from the communication region and added to d0 to specify the maximum number of lines.

An LFCB macro issued in the problem program updates the LTAB format.

Note: It is advisable to use the forms number parameter in conjunction with the linetab parameter. This causes the list writer task to pause, thus allowing the operator to mount the correct carriage control tape on the printer.

norbs Specifies the number of list records intercepted before the output is segmented (count-driven output segmentation). It is used only when spooling to disk intermediate storage. It is specified by one to six digits and overrides the POWER/VS generation specification of the RBS parameter of the POWER macro. A zero means that no segmentation will take place.

When output segmentation occurs, the operator is notified via message 1Q53I.

pnasename

Specifies the name of the phase under which an FCB (Forms Control Buffer) or UCB (Universal Characterset Buffer) image is cataloged in the core image library. Together with the UCB phase name, an option may be specified which can be either F, C, FC, or CF, where

F - signifies that the UCB is to be loaded with the folding operation code to permit printing of uppercase for lowercase bit configurations.

C - Prevents the data checks from being generated because of print line mismatches with the UCB.

The specified core image library phase is loaded into the FCB or UCB when a list writer task starts processing the output related to the * $$ LST statement.

Return to a user-default FCB or UCB can only be made via a subsequent * $$ LST statement.

When an * $$ LST statement with an FCB phase name specification is encountered by the execution processor, the specified buffer image is read to update the internal LTAB representation to reflect the new buffer. Any LTAB specification in the * $$ LST statement is therefore ignored if FCB is also specified. The FCB specification for a non-FCB printer is accepted, but is only used to update the LTAB. To maintain control of the buffer image when a job is cancelled, the * $$ LST statement is not flushed if the FCB or UCB is affected.

This parameter is not supported by the positional form of the * $$ LST statement.

lstaddr Specifies either the logical or physical address of the output printer for the list queue entry being created. The address can be either in the form SYSxxx, where xxx is LST or any valid programmer logical unit, or in the form cuu (or X'cuu'), which is the channel and unit number of an actual or a dummy printer. This parameter is needed when multiple printers are used to print multiple reports produced by a single program. If this parameter is omitted, the default is the first printer address specified at partition start-up. For example, if at start-up, the following printers were specified on SYSLOG: PRINTERS=00E,01E,02E, then printer 00E would be the default. In order to have the * $$ LST statement parameters take effect for printer 01E or 02E, LST=01E or LST=02E must be specified.

This parameter is not supported by the positional form of the * $$ LST statement.

* $$ PUN Statement

The * $$ PUN statement defines the attributes of the intercepted punch output of a POWER/VS job until either a subsequent * $$ PUN or an EOJ (* $$ EOJ or /&)

statement is encountered. Any number of * $$ PUN statements is permitted per POWER/VS job. If the * $$ PUN statement is not included in the job, the default values established during POWER/VS generation are assumed. * $$ PUN statements do not require that * $$ JOB and * $$ EOJ statements also be used.

For each * $$ PUN statement within a POWER/VS job, a separate punch queue entry is created (data-driven output segmentation).

Positional Format

```
* $$ PUN  [D                [A    ]]
          [disposition      [class]]
          [ƀƀƀƀ        ]
         ,[forms-number]
          [1               ]
         ,[number-of-copies]
          [tape-devaddr    ]
         ,[normbl]
```

Keyword Format

```
* $$ PUN [DISP={D           }]
         [     {disposition }]
         [,CLASS={A    }]
         [       {class}]
         [,REMOTE=remid]
         [,FNO={ƀƀƀƀ        }]
         [     {forms-number}]
         [,JSEP=sep]
         [,COPY={1               }]
         [      {number-of-copies}]
         [,TADDR=tape-devaddr]
         [,RBM=(norbm1, norbm2)]
         [,RBS=norbs]
         [,PUN=punaddr]
```

disposition

Specifies the disposition of the punched output after execution of the job. The disposition specification options are listed below. D, H, I, K, and L, are only valid when the output is spooled to disk. When the output is punched at a terminal, N and T are invalid.

D - or R - Delete after processing. The punch queue entry is automatically written to a punch according to its class and priority.

H - Hold. The punch queue entry remains in the punch queue; it is not written to a punch by POWER/VS until the operator changes the disposition to D or K by means of the PALTER command or issues the PRELEASE command.

I - Return output to input queue. In some situations it may be useful to return the punch output of a POWER/VS job directly to the input queue. For example, the assembler normally stores its output (including JCL) on tape, for subsequent input to a catalog run; under POWER/VS however, it can store output directly in the input queue.

The newly-built input queue entry has the same jobname, remote-id, and priority as the entry that produces the output. Class becomes that specified in the * $$ PUN statement; disposition becomes D. This facility should only be used for jobs producing punch output in executable format.

The I option is ignored in a writer-only partition. The option suppressess count-driven output segmentation and also ignores the multiple-copies specification.

K - Keep after processing. The output is automatically written to a punch according to its class and priority. When punching is complete, the punch queue entry is not deleted from the punch queue, and the disposition of the queue entry becomes L.

L - Leave in queue. The job remains in the punch queue; it is not written to a punch until the operator changes the disposition to K or D by means of the PALTER command or issues the PRELEASE command.

N - No spooling. If N is specified, punch requests are not intercepted. It allows direct writer capabilities without spooling. If the N option is specified and the device is not available at execution time, message 1Q46I is issued and punch output is spooled.

T - Tape spooling. The output is spooled to tape that is used as intermediate storage.

class

Specifies the class of the punch output of the job. It may be specified as any alphabetic character (A through Z). class may be used in conjunction with forms-number to group together the same types of punch output. Punch tasks can be specified by class to process the output. class is only meaningful when the output is spooled to disk.

If class is specified and the positional form is used, disposition must also be specified. (There is no comma between disposition and class in the positional form.) If class is omitted, class A is assigned. If the * $$ PUN statement is omitted, output class can be specified in the PSTART command.

remid

Specifies the TO remote-id to which punch output is to be routed. Each punch queue entry also has a FROM remote-id assigned to it, which identifies the user who initially submitted the job. remid can be a number from 0 to 200. Numbers 1 to 200 are specific remote users. If remid is specified as 0, punch output is directed to the central installation. If this parameter is omitted or if the positional form of the * $$ PUN statement is used, punch output is routed to the central location for locally submitted jobs or is routed according to the PUNROUT specification in the PRMT macro for jobs submitted remotely.

forms-number

Specifies the type of card forms to be used. forms-number is specified by one to four alphameric characters. If forms-number is not specified, it is assumed to be four blanks, representing the installation's standard type of card forms. When a punch writer task is to process output from a punch queue entry, it determines whether this forms-number matches that of the previous punch queue entry. If they match, the punch writer task processes the output. If they do not match, message 1Q40A is issued. If the first queue entry processed after a PSTART has a forms ID of four blanks, no mount message is issued.

sep

Specifies job separation. 0-9 can be specified. If 0 is specified, no job separation occurs. Job separation means that before the job's punched output one card containing the POWER/VS jobname (punched so that it can be read on the back of the card) and from two to eight cards containing 12-11-0-8-9 punches (in all 80 columns) are added and that behind the job's punched output two blank cards are added. This occurs if 1, 2, or 3 is specified. If 4 is specified, one additional 12-11-0-8-9 card is punched; if 5 is specified, two additional 12-11-0-8-9 cards are punched, and so on.

Note: Stacker selection is ignored if job separation is requested. The default stacker for the given device is used instead.

number-of-copies

Specifies the number of copies to be punched from the punch queue entry. The parameter is specified by one or two digits. If it is not specified, 1 is assumed. If 0 or 00 is specified, one copy is punched. This parameter is ignored when the output is not spooled to disk. If additional copies are required when disposition is T, the tape must be repunched.

tape-devaddr

Specifies the hardware address of the tape drive to be used for spooling when the job is executed. Its format is one of the following:

```
cuu
X'cuu'
(cuu,X'ss')
(X'cuu',X'ss')
(cuu,ss)
(X'cuu',ss)
```

where X'ss' specifies the density of the output tape. tape-devaddr forces disposition T if invalidly specified. If DISP=T and tape-devaddr is omitted, the operator is requested via message 1Q55D to specify the tape addres at job execution time.

Multiple files can be written to a single tape. Multivolume files are also supported. No label checking is performed.

norbm1 Specifies the number of punch records intercepted before message 1Q52I is sent to the operator. This parameter is specified by one to six digits. If norbm1 is omitted, the value defined during POWER/VS generation in the first entry of the STDCARD parameter of the POWER macro is assumed.

norbm2 Specifies the additional number of punch records intercepted before message 1Q52I is issued again. It is specified by one to six digits. If norbm2 is omitted, the preceding comma as well as the parentheses may be omitted and the value defined during POWER/VS generation in the second entry of the STDCARD parameter of the POWER macro is assumed. This valvue is also assumed if the positional form of the * $$ PUN statement is used.

norbs Specifies the number of punch records intercepted before the output is segmented (count-driven output segmentation). It is used only when spooling to disk intermediate storage. It is specified by one to six digits and overrides the POWER/VS generation specification of the RBS parameter of the POWER macro. A zero means that no segmentation will take place. When output segmentation occurs, the operator is notified via message 1Q53I.

punaddr Specifies either the logical or physical address of the output punch for the punch queue entry being created. The address can be either in the form SYSxxx, where xxx is PCH or any valid programmer logical unit, or in the form cuu (or X'cuu'), which is the channel and unit number of an actual or a dummy punch. If this parameter is omitted, the default is the first punch address specified at partition start-up. This parameter is needed when multiple punches are used to punch multiple card files produced by a single program.

* $$ RDR Statement

The * $$ RDR statement is used to insert a 3540 diskette file into input being read from the card reader. There are two modes of input processing, that is

1. SYSIN-mode processing; the physical unit number of the 3540 diskette device need not be specified. The file to be read from the diskette must contain the necessary JECL and/or JCL statements.

2. Data-mode processing; the physical unit number of the diskette from which the input data file is read must be specified. The JECL and/or JCL statements are read from the card reader. If multiple RDR statements are issued in one POWER/VS job, they must specify the same physical unit number.

Positional Format

```
* $$ RDR  [physical-unit-number]
         ,['file-id']
          [1                  ]
         ,[number-of-diskettes]
         ,[S]
```

Keyword Format

```
* $$ RDR [DEV=physical-unit-number]
         [,FID='file-id']
         [     {u                  }]
         [,NOD={number-of-diskettes}]
         [,VSC={NO }]
         [     {YES}]
         [,VER={NO }]
         [     {YES}]
```

physical-unit-number

Specifies the physical unit number (X'cuu' or cuu) used by the user program to read the input data file. The physical unit must not be specified for SYSIN files. In one POWER/VS job more than one RDR statement can be specified to insert more than one 3540 data file, but the different RDR statements must indicate the same physical unit number in the DEV parameter.

'file-id'

Specifies the file name (in quotes) as it appaers in the HDR1 label on the 3540 diskette. One to eight alphameric characters can be specified. If this parameter is omitted, POWER/VS read the file specified by the first non-secured HDR1 label found on the diskette that is currently mounted. The size of the record to be read is taken from the HDR1 label.

number-of-diskettes

Specifies the maximum number of diskettes to be read. One to three digits can be specified. Allowed values are 1 to 255. The default value is 1. Reading stops after the specified number of diskettes are processed, or after reading a diskette whose HDR1 label indicated that it was the last volume of the file.

VSC={NO
YES} Specifies whether volume sequence checking is desired. If omitted, no sequence checking will take place. Sequence numbers must start with 1 and must be incremented by 1 up to a maximum of 99.

S See the description for VSC= above.

VER={NO
YES} If VER=YES is specified, POWER/VS checks that file verification had been performed on the diskette file (verify field in the HDR1 label of the file set to 'V'). If VER=YES and the verify field was not set to 'V', the file is rejected at open time. If VER=NO or if the parameter is omitted, the verify field in the HDR1 label is ignored during processing.

* $$ SLI Statement

At execution time the * $$ SLI statement causes the private and the system source statement library to be searched for a book, which, when found, is inserted into the job stream. The private source statement library, if assigned, is searched first.

An * $$ SLI statement does not require that * $$ JOB and * $$ EOJ statements also be used.

Statements may be added to or deleted from the source statement library books based on codes in columns 73-80 of the input stream statements. Statements from the reader queue containing $SLI in columns 73-76 have a special meaning and are called source library (SL) update statements. They may be inserted before or after SSL statements or may replace or delete them.

Columns 78-80 of an SL update statement contain a sequence number. Column 77 contains a special code:

A - insert after book statement with same sequence number

B - insert before book statement with same sequence number

D - delete the book statement with the sequence number.

Other - replace the book statement with same sequence number of insert this statement.

SL update statements and * $$ DATA statements, when used, must follow the * $$ SLI statement in the same sequence as the statements in the book they correspondent to. SL update statements must contain $ SLI in columns 73-76.

Books are cataloged into the source statement library with the DOS/VS program MAINT. To include /* and /& statements in a book, submit them as * $$/* and * $$/& statements since /* and /& statements are not accepted as data by MAINT. If the book is not found in the specified sublibrary, message 1Q44I is issued and the job is flushed.

Format

* $$ SLI [sublib.]bookname

sublib Specifies the name of the sublibrary to be searched. Valid sublibrary names are A through Z, 0 through 9, $, #, or @. Sublibraries A, C, D, E, F, and Z are already defined as special purpose libraries in DOS/VS. This parameter is optional. If omitted, the SUBLIB specification at POWER/VS generation is used.

bookname Specifies the book name to be searched for and inserted into the job stream.

Delimiter Statements

The end-of-job step and end-of-job statements are used to delimit DOS/VS job definitions within the SSL books. (Refer to * $$ SLI statement.)

Format

* $$/* (end-of-job step)
* $$/& (end-of-job)

POWER/VS Central Operator Commands

Communication between POWER/VS and the central operator is established via the attention routine (AR). Pressing the request key causes the message:

AR 1I60A READY FOR COMMUNICATIONS

to be displayed on SYSLOG. Hereafter, any POWER/VS central operator command can be entered. (For the display operator console on a System/370 Model 115 or Model 125, central operator commands can be entered directly without pressing the request key.)

Functions of the Central Operator Commands

The central operator commands include:

Task management commands Control reader/writer tasks and execution processors.

Queue management commands Control the various input/output queues.

Miscellaneous commands Align printer forms or save the POWER/VS account file.

Format of the Central Operator Commands

A central operator command has two fields, the operation field and the operand field. The operation field can be specified in either the extended or the abbreviated format. The operand field contains one or more parameters separated by commas, or contains no parameters at all. If operands are specified, they must be separated from the operation field by at least one blank. If an optional parameter is omitted, its delimiting comma must be coded, unless it is the last one specified in the operand field. The operator commands can be entered in either uppercase or lowercase.

Description of the Central Operator Commands

POWER/VS supports abbreviated as well as extended operation codes. All operands (parameters) are valid for both formats. Figure 9 shows the abbreviated and the extended codes:

Type	Extended Format	Abbreviated Format	Function
Task management	PSTART	S	Start a task or partition.
	PSTOP	P	Stop a task or partition.
	PGO	G	Activate a task or partition.
	PEND[1]	-	End POWER/VS execution.
	PCANCEL	C	Cancel a POWER/VS status report.
	PFLUSH	F	Flush an active job.
	PRESTART	T	Restart a writer task.
Queue management	PDISPLAY	D	Display job status.
	PALTER	A	Alter attributes.
	PDELETE	L	Delete a job or message.
	PRELEASE	R	Release a job.
	PBRDCST	B	Transmit a message.
	PINQUIRE	I	Check terminal status.
	PACCOUNT	J	Process account file.
	PSETUP	-	Print page layout.

[1] No abbreviated format of the PEND command is provided since the operator might inadvertently terminate POWER/VS operation.

Figure 9. POWER/VS Central Operator Commands.

The following pages describe all the central operator commands in alphabetical order. A summary of these commands is contained in Appendix C.

PACCOUNT Command

The PACCOUNT command can be used at any time to save the accumulated account file records. The operator is notified by a warning message when the account file becomes 80% full.

If any POWER/VS task requires an account record and there are no free records, a message is issued, and the task requesting the account record is placed in the wait state until the account file is emptied by the PACCOUNT command.

After the message 'ACCOUNT FILE SAVED', the entire DASD space allocated to the file is available for more account records. It is the user's responsibility to process the supplied account information.

If no operand is specified, the account file is spooled to disk. It is stored with priority 1 in output class P with DISP=H in the punch queue. In order to get the account information punched, the operator may start a punch writer task on output class P.

Format

```
{PACCOUNT}  [tapeaddr[,filename]]
{J       }  [DISK,filename      ]
            [DEL                ]
```

tapeaddr Specifies the tape on which the file is to be written. Its format is:

```
cuu
X'cuu'
(cuu,ss)
(cuu,X'ss')
(X'cuu',ss)
(X'cuu',X'ss')
```

where cuu is the channel and unit number and where ss specifies the density of the tape.

Before any record is written to the specified tape, label checking is performed.

filename If filename is specified with tapeaddr, the file created will be a standard labeled tape file. filename is from one to seven alphameric characters long. A standard label for the tape file must be present on the label information cylinder.

If filename is not specified, the file is a standard unlabeled DOS/VS tape file containing variable-length records.

DISK,filename
Specifies that the POWER/VS account file is to be written to disk. filename is from one to seven alphameric characters long. It is the user's responsibility to supply // DLBL and // EXTENT statements (for a sequential disk output file) on the label information cylinder. The disk file created is a copy of the IJAFILE.

DEL Specifies that the account file records are to be deleted.

Examples:

PACCOUNT	280	Save the account file on an unlabeled tape mounted on unit X'280'.
PACCOUNT	DEL	Delete all account file records.
PACCOUNT		Spool account file records to punch queue.

PALTER Command

The PALTER command is used to change the various parameters of a queue entry that is already queued. Any number of parameters may be modified with one command.

If an input queue entry is specified and an execution processor is currently processing this entry, message 1R88I is issued.

The priority of an entry in the input queue is identical to that of the corresponding entries in the output queues. On the other hand, the disposition and class may be totally different in corresponding input and output queue entries, since input disposition and class are defined in the * $$ JOB statement, whereas output disposition and class are defined in the * $$ LST and * $$ PUN statements.

The COPY and REMOTE parameters apply to output only. If applied to an input queue entry they are ignored. If an output queue entry is specified in the PALTER command and a writer task is currently processing the entry, only the number of copies may be altered. If any other parameter is specified, message 1R88I is issued and the command is ignored.

PDISPLAY may be used to check the results of the PALTER command.

Format

```
{PALTER}queue, {jobname[,jobnumber]}
{A     }       {ALL                }
               {*abc               }
               {class1             }
               [,PRI=priority]
               [,DISP=disposition]
               [,CLASS=class2]
               [,COPY=number-of-copies]
               [,REMOTE=remid]
```

queue Specifies the queue for which the command is to be executed:

LST = list queue
PUN = punch queue
RDR = reader queue

jobname Specifies the name by which the job entry is known to POWER/VS. It can be two to eight alphameric characters.

jobnumber
Specifies the number assigned to the job entry by POWER/VS. It may be one to five digits long. The PDISPLAY command may be used to obtain the correct specification for the jobnumber. This number can be used to identify different jobs with the same jobname.

ALL
Requests that all entries in the specified queue are to be altered.

*abc
Requests to alter all job entries that have the first characters of their jobnames in common. abc represents any combination of from one to seven alphameric characters.

class1
All job entries of a certain class are to be altered in the specified queue.

priority
Specifies the priority of the job entry. It is specified as a single digit from 0-9. Nine is the highest priority.

disposition
Can be specified as H for hold, K for keep after processing, L for leave, or D for delete after processing.

class2
Can be specified as any alphabetic character from A through Z. Also specifications from 0 through 4 are allowed for input classes.

remid
Can be specified as any number from 0 to 200. 0 indicates the central location. Only the TO remote-id can be modified with a PALTER command; the FROM remote-id cannot be modified.

number-of-copies
Specifies the number of copies to be produced after the command is issued, including the one in progress. It can be 0 through 99.

Examples:

```
PALTER LST,PAYROLL,DISP=H,CLASS=B
```
Alter the disposition of the output of job PAYROLL to H, and alter the class to B.

```
A PUN,*PAYR,DISP=D
```
Alter the disposition of all jobs that begin with PAYR to D.

```
ALTER LST,P,CLASS=Q
```
Change all class P jobs to Q.

PBRDCST Command

The PBRDCST command enables the central operator to communicate with remote users. Messages can be submitted for one specific user or for all users.

ALLUSERS-type messages (remid=ALLUSERS) are queued in virtual storage and are displayed at the user's request only (DISPLAY command).

Broadcast messages that are routed to one specific operator are displayed as soon as the terminal is ready to receive the message.

Format

```
{PBRDCST}
{B      } remid,'text'
```

remid
Specifies the remote-id to which the command applies. Remote users can be identified by numbers from 1 to 200. A specification of 0 indicates the central location, and a specification of ALLUSERS indicates all users.

text
Specifies the text of the message. The message can consist of up to 59 characters enclosed in single quotation marks. A single quotation mark within the message must be written as two quotation marks.

Examples:

```
PBRDCST 150,'RJE will be shut down
 in 30 min'
```
Broadcast the message to the user whose remote-id is 150.

```
PBRDCST ALLUSERS,'service resumes
 at 8.00 a.m.'
```
Broadcast the message to all users in the system.

PCANCEL Command

The PCANCEL command is used to terminate the printing initiated by the PDISPLAY command. A confirmation message is issued.

Format

```
{PCANCEL}
{C      } [STATUS]
```

STATUS
This operand is optional.

Example:

PCANCEL
Cancels printing initiated by PDISPLAY.

PDELETE Command

The PDELETE command is used to remove one or more queue entries from the specified queue. Queue entries being processed by an execution processor or writer task are not affected by this command.

Format

```
{PDELETE} [queue,{jobname[,jobnumber]}]
{L      } [      {ALL                }]
          [      {class              }]
          [      {*abc               }]
          [MSG[,n]                     ]
```

queue Specifies the queue for which the command is to be executed:

LST = list queue
PUN = punch queue
RDR = read queue

jobname Specifies the name by which the job entry is known to POWER/VS. jobname can be two to eight alphameric characters.

jobnumber Specifies the number assigned to the job entry by POWER/VS. The jobnumber is from one to five digits long. PDISPLAY can be used to obtain the jobnumber.

ALL All queue entries are to be deleted from the specified queue.

class All queue entries of a certain class are to be deleted from the specified queue.

*abc All queue entries with the same first characters are to be deleted. abc represents any combination of up to seven alphameric characters.

MSG,n ALLUSERS-type message number n is to be deleted. The actual message number can be obtained with a PDISPLAY MSG command. If n is omitted, all ALLUSERS-type messages issued by the central operator are deleted.

Examples:

PDELETE RDR,ALL
Delete the entire read queue.

PDELETE LST,B
Delete all entries in the list queue with a class of B.

PDELETE MSG
Delete all ALLUSERS-type messages issued by the central operator.

PDISPLAY Command

The PDISPLAY command provides the operator with a status report on SYSLOG of the queue entry or entries specified by the operand. Each queue entry is represented by one line containing the following items.

Items	Characters
jobname	8
jobnumber	5
priority	1
disposition	1
class	1
FROM remote-id	3
TO remote-id	3
Number-of-records/pages	6
Number-of-copies	2
Forms-number	4

The last four items are displayed for output only. For punch output the number of records is displayed, whereas for list output the number of pages is displayed.

Each line of the status report starts with the message number 1R46I. If the status report is made up of entries from different queues, the name of the queue is printed before the first entry is displayed.

Format

```
{PDISPLAY}  ( queue,jobname[,jobnumber] )
{D       }  ( queue[,ALL]               )
            ( queue,HOLD                )
            ( queue,FREE                )
            ( queue,RJE[,remid]         )
            ( queue,LOCAL               )
            ( queue,*abc                )
            { queue,class               }
            { ALL[,listaddr]            }
            ( HOLD                      )
            ( FREE                      )
            ( RJE[,remid]               )
            ( LOCAL                     )
            ( *abc                      )
            ( MSG                       )
            ( A                         )
            ( M                         )
            ( Q                         )
```

The disposition of a queue entry may appear as H for hold, K for keep after processing, L for leave, D for delete after processing or * if an execution processor or a writer task is currently processing the entry.

The class can appear as an alphameric character from A to Z, or from 0 to 4.

Remote-id is displayed as a number from 0 to 200. The remote-id of the central location is 000.

If a writer task is processing a queue entry when the PDISPLAY command is entered, number-of-copies is the number of copies left to be produced, including the one in progress. The number-of-records/pages is the number of records or pages left to be produced. The operator can use this information to determine whether to stop a writer task or let it continue until the current queue entry is completed.

queue Specifies the queue for which the PDISPLAY command is to be executed:

LST = list queue
PUN = punch queue
RDR = read queue

jobname Specifies the name by which the job entry is known to POWER/VS. The jobname can be two to eight alphameric characters.

jobnumber

Specifies the number assigned to the job entry by POWER/VS. The jobnumber can be one to five digits.

ALL Specifies a request for the status of all queue entries in the specified queue. If the queue parameter is not specified, status information on all entries in all queues is displayed on SYSLOG.

listaddr

Specifies the printer to which the status report is to be sent. Its format is:

cuu or X'cuu'

HOLD Specifies a request for the status of all entries in the specified queue that are not available for processing (disposition H or L). If the queue parameter is not specified, status information is displayed on all such queue entries.

FREE Specifies a request for the status of all entries in the specified queue that are available for processing (disposition D or K). If the queue parameter is not specified, status information is displayed on all such queue entries.

RJE Specifies a request for the status of all RJE-type entries in the specified queue. If the queue parameter is not specified, status information is displayed on all such queue entries.

RJE,remid

Specifies a request for the status of all entries in the specified queue that are submitted by or routed to a specific remote user. If the queue parameter is not specified, status information is displayed on all such queue entries.

LOCAL Specifies a request for the status of all entries in the specified queue that are submitted from or routed to the central location. If the queue parameter is not specified, status information is displayed on all such queue entries.

class Specifies a request for the status of all job entries in the queue with the specified class. class can be specified as a character from A through Z, or from 0 to 4 (input class only).

*abc Specifies a request for the status of all entries having the first characters of their jobnames in common. abc represents any combination of from one to seven alphameric characters. If the queue parameter is not specified, status information is displayed for all queues. The asterisk (*) must be keyed.

MSG Specifies a request to display all ALLUSERS-type messages that have been entered together with their originators. The number displayed with each message can be used to delete that message.

A Specifies a request for a list of all active reader/writer tasks together with the job entries they are currently processing.

M Specifies a request to redisplay a list of all POWER/VS messages for which an operator reply is still required.

Q Specifies a request for the number of free queue records, and free POWER/VS account records (if applicable).

Examples:

Display the status of all entries in all queues.

```
AR PDISPLAY ALL
F1 1R46I  READER QUEUE   P D C                 *****
F1 1R46I  DISKDUMP 00068 9 L A                 *****
F1 1R46I  PAUSE    00070 9 L A                 *****
F1 1R46I  DITTO    00071 3 L A                 *****
F1 1R46I  PRINTLOG 00069 1 L A                 *****
F1 1R46I  LIST QUEUE     P D C  PAGES CC FORM  *****
F1 1R46I  PAUSE    00070 9 D A      1  1       *****
F1 1R46I  PAYINPUT 00063 3 D A      3  1       *****
F1 1R46I  PAYEDIT  00064 3 D A     75  1       *****
F1 1R46I  PAYSORT  00065 3 D A      5  1       *****
F1 1R46I  PAYCARDS 00066 3 D A      7  1 CHEX  *****
F1 1R46I  PAYLIST  00067 3 D A     44  1       *****
F1 1R46I  PRINTLOG 00069 1 D A      5  1       *****
F1 1R46I   PUNCH QUEUE   P D C  CARDS CC FORM  *****
F1 1R46I  PAYCARDS 00066 3 D A    233  1       *****
```

Display the status of all jobs that begin with PAY.

```
AR PDISPLAY *PAY
F1 1R46I  READER QUEUE NOTHING TO DISPLAY      *****
F1 1R46I    LIST QUEUE   P D C  PAGES CC FORM  *****
F1 1R46I  PAYINPUT 00063 3 D A      3  1       *****
F1 1R46I  PAYEDIT  00064 3 D A     75  1       *****
F1 1R46I  PAYSORT  00065 3 D A      5  1       *****
F1 1R46I  PAYCARDS 00066 3 D A      7  1 CHEX  *****
F1 1R46I  PAYLIST  00067 3 D A     44  1       *****
F1 1R46I   PUNCH QUEUE   P D C  CARDS CC FORM  *****
F1 1R46I  PAYCARDS 00066 3 D A    233  1       *****
AR PRELEASE RDR,PAUSE
F1 1R88I  CP READY
F1 1Q47I   BG PAUSE   -00070 FROM 000
BG // JOB PAUSEJOB
```

The lines of the status report are not chained, and can, therefore, be interspersed with other DOS/VS messages.

PEND Command

The PEND command is used to terminate POWER/VS. Unless KILL is specified, all active tasks are allowed to continue until the queue entry they are currently processing is completed. POWER/VS controlled partitions are released as soon as the job corresponding to the current input queue entry is terminated. After all supported partitions are released and all readers and writers have stopped, the POWER/VS partition is released, and the system is restored for normal DOS/VS operation. If KILL is specified, POWER/VS is canceled immediately.

After abnormal termination of POWER/VS, the printer remains assigned in the partition.

POWER/VS acknowledges the PEND command with a message.

Format

```
PEND [uraddr           ]
     [KILL[,uraddr]    ]
```

KILL Terminates POWER/VS immediately for emergency reasons. Partitions supported by POWER/VS are also canceled.

uraddr Specifies the unit record device address of the printer on which the status report is to be printed. A dump is printed if KILL is specified. The dump cannot be canceled. If uraddr is not specified no status report or dump is produced.

uraddr can be entered in either of two formats:

cuu or X'cuu'

Examples:

PEND KILL,00E
Emergency termination with system dump.

PEND Normal termination. No status report, no system dump.

PEND 00E Normal termination with status report. No system dump.

PFLUSH Command

The PFLUSH command is used to cancel the current job entry being created or processed by a POWER/VS task. The PFLUSH command is ignored if issued in a writer/only partition.

When a reader task is flushed, further input processing is terminated for the current entry, and no queue entry is made.

The specified reader/writer task remains active and continues by processing the next appropriate entry in the input/output queue.

For a partition, the current DOS/VS job is canceled and any other DOS/VS jobs in the same POWER/VS job are flushed. Also, the corresponding reader entry is deleted (if HOLD is not specified). A list entry is created that contains a message to the programmer stating that his job has been flushed by the operator. Processing continues with the next appropriate entry in the reader queue.

Format

```
{PFLUSH}   {uraddr[,HOLD]     }
{F     }   {partition[,HOLD]  }
```

uraddr Specifies the unit record device associated with the reader/writer task. Its format is:

cuu or X'cuu'

HOLD Specifies that the corresponding queue entry is not to be deleted, but put in the hold state; that is, the input queue entry must not be deleted when a partition is flushed, or the output queue entry must not be deleted when a writer task is flushed. If this parameter is specified when flushing a reader task, it is ignored.

The HOLD option is useful when a job needs more devices or other resources than are currently available. The job can then be rescheduled when the resources become available. Jobs requiring the same resources can be grouped together in input or output classes to prevent such inconvenience.

partition
Specifies the partition in which the job to be flushed is executing as: BG, F4, F3, F2, or F1, and the partition is canceled.

Examples:

PFLUSH 00E
Delete list queue entry currently being printed. Continue with next entry.

PFLUSH BG
Terminate processing of current read queue entry. Delete this entry. Continue with next entry.

PFLUSH BG,HOLD
Same as preceding example, except that the read entry is preserved and put in the hold state.

PGO Command

The PGO command reactivates a POWER/VS task that is waiting for an operator response. Writer tasks are reactivated by using the uraddr parameter. Execution processors are reactivated by the partition,cuu parameter.

The PGO command is normally used after an action-type message is issued to the operator from POWER/VS. It is not used to restart partitions or tasks stopped by a PSTOP command.

Note: Reader tasks, operating on card readers that give a device end interrupt when they are readied, are automatically reactivated when the device end occurs. They do not need a PGO command.

Format

$$\left\{\begin{array}{l}\text{PGO}\\\text{G}\end{array}\right\}\left\{\begin{array}{l}\text{uraddr}\\\text{partition,cuu}\end{array}\right\}$$

uraddr Specifies the unit record device to be reactivated. It has the format:

cuu or X'cuu'

partition,cuu
Specifies the partition and unit record device address whose output is being spooled to tape. The partition can be: BG, F4, F3, F2, or F1.

Examples:

PGO 00E Reactivate list task after, for example, aligning printer forms.

PGO BG,00E
Reactivate BG execution after mounting a new spool tape for printer output.

PINQUIRE Command

The PINQUIRE command provides the central operator with a report for a specific RJE line or for all lines. The status can be: processing, inactive, not initiated, or not supported.

processing
The remote user has entered a valid SIGNON command. In this case, the remote-id is displayed together with the line status.

inactive
The line has been started by the central operator with a PSTART command, but no user is currently signed on.

not initiated
The line has not been started by the central operator.

not supported
During POWER/VS generation, no PLINE macro was specified for this line.

Format

$$\left\{\begin{array}{l}\text{PINQUIRE}\\\text{I}\end{array}\right\}\left\{\begin{array}{l}\text{lineaddr}\\\text{ALL}\end{array}\right\}$$

lineaddr Specifies the line address associated with the RJE task. Its format is:

cuu or X'cuu'

ALL The status of all supported lines will be displayed.

Example:

PINQUIRE ALL
Display the status of all supported lines.

```
030   PROCESSING 022
031   PROCESSING 150
032   PROCESSING 004
033   NOT INITIATED
034   INACTIVE
```

PRELEASE Command

The PRELEASE command is used to take one or more job entries out of the hold or leave state and make them available for processing. After processing, a job entry originally in the leave state returns to the leave state; a job entry originally in the hold state is deleted from the queue.

If a PRELEASE command is issued for a queue entry that is not in the hold or leave state the command is ignored. A job entry that is to be printed or punched at a terminal cannot be released by the central operator.

Format

```
{PRELEASE} ,queue,jobname[,jobnumber]
{R       } {queue[,ALL]               }
           {queue,class               }
           {queue,*abc                }
```

queue Specifies the queue for which the command is to be executed:

LST = list queue
PUN = punch queue
RDR = reader queue

jobname Specifies the name by which the job entry is known to POWER/VS. jobname may be two to eight alphameric characters.

jobnumber Specifies the number assigned to the job entry by POWER/VS. jobnumber is one to five digits long. The PDISPLAY command can be used to obtain the number.

ALL All job entries in the specified queue are to be released.

class All job entries of a certain class in the specified queue are to be released.

*abc All job entries with the same first characters are to be released. abc represents any combination of up to seven alphameric characters.

Example:

PRELEASE LST,PAYROLL
Release the entry PAYROLL from the list queue.

PRELEASE RDR,ALL
Release the entire read queue.

PRELEASE LST,*PAY
Release all entries in the list queue whose names begin with PAY.

PRESTART Command

The PRESTART command directs the specified list/punch task to stop and tc resume printing/punching either from the beginning of the list/punch entry, or up to 9999 pages (or cards) forward or backward from the point of interruption.

The number of duplicate pages or cards written because of a PRESTART command is separately counted and stored in the list/punch account record. A page is defined as a skip-to-channel-one.

If the user specifies too large a backward count, or if the operand is omitted, printing/punching will resume from the first page/card. If the user specifies too large a forward count, message 1Q42 is issued.

If PRESTART is used to space forward for printing the end of a job first (for example, to print totals), then the disposition of the queue entry should first be altered to L in order to prevent the print job from being deleted from the queue.

Format

```
{PRESTART} uraddr[,n]
{T       }
```

uraddr Specifies the unit record device address associated with the writer task. Its format is:

cuu or X'cuu'

n Specifies a signed or unsigned value from 0 to 9999. A plus sign indicates a forward count, a minus sign indicates a backward count from the last interrupted page/card. No sign indicates a count from the beginning.

Example:

PRESTART 00E,-10
Reprint the previous ten pages and continue to end of output.

PSETUP Command

The PSETUP command can be entered only after message 1Q40A, requesting a special form to be mounted, is displayed. It causes n pages of spooled list data to be printed. All printable characters of the page are replaced by the character X. During the printing of the n pages, the operator can manually align the printer forms and can reissue PSETUP as many times as is necessary.

The number of pages printed by PSETUP will be counted and stored separately in the list account record.

When the PGO command is issued to resume printing, the pages are printed again with their original contents.

Format

PSETUP   uraddr[,n]

uraddr   Specifies the unit record device associated with the write task. Its format is cuu or X'cuu'

n   Specifies, as one or two digits, the number of pages to be printed. If n is omitted, only one page is printed.

Example:

PSETUP 00E,2
         Print two pages to allow forms alignment.

Then, PGO 00E
         Pages are printed again with their original contents. Printing continues.

```
                                                                            xxxx xxx   xx

         xxx xx xxx xxxxx xxxxxxxxxxxxxxx xxxxxxxx                               xxxxxxxxx
1st
Page    xxx xxxxxxx xxxxxxxxxxxx xxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxx xxxxxxx

              x xxxx



                                                                            xxxx xxx   xx

        xxx xx xxx xxxxx- xxxxxxxxxxxxxxxxxxx xxxxxxxx                          xxxxxxxxx
2nd
Page       xxxx xxxxxxx xxxxxxxxxxx xxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxx xxxxxxx

              x xxxx



                                                                               SEPT 30,   74

         PAY TO THE ORDER OF.........ONE EMPLOYEE                                ***123.45
1ST
Page        ONE HUNDRED TWENTY-THREE AND ------------------------------ 45/100 DOLLARS

                * 2001



                                                                               SEPT 30,   74

         PAY TO THE ORDER OF..........ANOTHER EMPLOYEE                           ***543.21
2nd
Page        FIVE HUNDRED FORTY-THREE AND ------------------------------ 21/100 DOLLARS

                 # 2002
```

PSTART Command

This command starts a POWER/VS task or brings a partition under control of POWER/VS.

The task and RJE formats are used to start reader/writer tasks and RJE tasks, respectively. The partition format is used to place a partition under POWER/VS control.

POWER/VS requires a programmer logical unit for each unit record device, each spool tape, or each RJE line specified in a PSTART command. If no free LUB is available in the POWER/VS partition, a message is displayed and the command is ignored.

Format

For the special format of the PSTART command that applies to diskette RDR tasks, refer to Format of PSTART Command for 3540 Reader Tasks, later in this section.

```
{PSTART} ┌task,uraddr[,class][,bufno]          ┐
{S     } │task,uraddr,tapeaddr                 │
         │partition[,class][,outputclass]      │
         └RJE,lineaddr[,password]              ┘
```

task
Specifies the task to be started as RDR, LST, or PUN.

RDR = reader task
LST = list task
PUN = punch task

When a reader task is started, input entries are built from the input data. When a writer task is started, output entries are processed in priority within class.

uraddr
Specifies the unit record device address associated with the reader/writer task. Its format is either:

cuu or X'cuu'

bufno
This parameter applies only to card-reader tasks and specifies the number of buffers used to read in cards. 1 or 2 can be used. If omitted, 1 is assumed.

lineaddr
Specifies the line address associated with the RJE task. Its format is either:

cuu or X'cuu'

tapeaddr
Specifies the address of the tape unit on which a spool tape is mounted. Its format is:

X'cuu'

Only list and punch tasks can be started from a tape. If the tape is not recognized as a spool tape, a message is issued and the command is ignored.

password
When an RJE line is started a password can be set for the line by the central operator. This password can be any combination of up to eight alphameric characters. A default password is defined during POWER/VS generation, and is used if no password is specified by the central operator. If the password is specified in the PSTART command, this password is in effect until a PSTOP command is issued. If a password is set, a remote operator must use it to sign on.

class
The meaning of this parameter depends on the type of task to be started.

For a writer task, class defines the output class(es) upon which the task operates. Up to four classes can be designated by specifying one to four alphabetic characters from A through Z. The order specified is the order in which the classes will be processed. If no class parameter is specified, only class A is selected.

For a reader task, class defines the input class that is assigned to all jobs without a class specification in their * $$ JOB cards, when no * $$ CTL statement is in effect. It may be specified as an alphameric character from A through Z or from 0 through 4. If no class parameter is specified, class defaults to A.

For a partition, class defines the input class(es) that can be executed in this partition. Up to four classes can be designated by specifying one to four alphameric characters from A through Z or from 0 through 4. The order specified is the order in which the classes will be executed. If no class is specified, only job entries with matching partition-type (0-4) input class are selected.

Note: Only one of the classes specified in the partition form of a PSTART command can be a partition-type class (that is, 0-4), and the partition-type class specified must match the partition being started. For example, PSTART F2,AC2.

partition
Specifies the partition to be brought under POWER/VS control as:

BG, F4, F3, F2, or F1

If the partition specified is already controlled by POWER/VS, the PSTART command acts as a normal AR START. If the partition is not under POWER/VS control, the PSTART command first brings it under control of POWER/VS and then performs the AR START function.

If the partition to be placed under POWER/VS control is still active, or not supported by the DOS/VS system, or if it does not have lower priority than the POWER/VS partition, a message is issued and the command is ignored.

outputclass
This parameter can be used to specify the class of a POWER/VS job's printed or punched output if * $$ LST or * $$ PUN statements were omitted. It can be any alphabetic character.

Examples:

PSTART RDR,00C,2
Start reader task on X'00C'. Assign class 2 to all jobs without specific class assignment.

PSTART LST,00E,AB
Start list task for class A and B.

S PUN,00D,X'285'
Start tape punch task on X'00D' from tape mounted on 285.

PSTART BG
Bring BG under control of POWER/VS. Select input class 0 only.

PSTART F1,AB
Bring F1 under control of POWER/VS. Select input classes A and B only.

PSTART RJE,030,TORONTO
Start RJE task on line 030. TORONTO is the security password.

Format of PSTART Command for 3540 Reader Tasks

The following format of the PSTART command is to be used to start a reader task that merges a 3540 diskette file with input being read from the card reader.

```
{PSTART}RDR,uraddr[,class],uraddr
{S     }
```

where the first physical unit number (uraddr,X'cuu' or cuu) is the address of the card reader and the second one the address of the 3540 diskette.

The format of the PSTART command to start a reader task that reads both job control statements and data from a 3540 diskette is as follows:

```
{PSTART}RDR,uraddr[,class][,'file-id']
{S     }                  [, file-id  ]
[,1                  ]
[,number-of-diskettes][,S][,V]
```

uraddr
Specifies the physical unit number (X'cuu' or cuu) of the 3540 diskette.

'file-id'
file-id
Specifies the file name as it appears in the HDR1 label on the 3540 diskette. This name must be idtentical with the file identifier, passed in the DLBL statement when the file was created. Blank characters are allowed in the file name if it is specified in quotes.

number-of-diskettes
Specifies the maximum number of diskettes to be read. One to three digits can be specified. Allowed values are 1 to 255. The default value is 1. Reading stops after the specified number of diskettes are processed, or after reading a diskette whose HDR1 label indicated that it was the last volume of the file.

S
Specifies whether volume sequence checking is desired. If omitted, no sequence checking will take place. Sequence numbers must start with 1 and must be incremented by 1, up to a maximum of 99.

V Specifies that a check on file verification is requested. If file verification had been performed on the diskette file, the verify field in the HDR1 label was set to 'V'. If 'V' is specified and the verify field in the HDR1 label was not set to 'V', the file is rejected at open time. If 'V' is not specified, the verify field in the HDR1 label is ignored during processing.

Example:

Job control statements and data are both on one 3540 diskette file (SYSIN).

```
   PSTART RDR,X'00B',B,'MYFILE',31
```

Start reader task on X'00B'. Reading stops after 31 diskettes are processed or after reading a diskette that does not have a continuation indicator in its HDR1 label. Assign class B to all jobs without specific class assignment.

PSTOP Command

The PSTOP command terminates a POWER/VS task or releases a partition from control of POWER/VS.

When a reader task is stopped, further input processing is terminated, and the current reader entry is not placed in the input queue.

When a writer task is stopped, further output processing is terminated, but the current job entry segment is not deleted from the output queue. When a writer task is restarted, processing of the entry starts either at the first record or the next record of the same job entry segment, or starts by processing the next job entry, depending on the parameters specified in the PSTOP command.

For reader/writer or RJE tasks, POWER/VS acknowledges the PSTOP command with a message at both the central location and at the terminal.

Format

```
{PSTOP}  {uraddr[,EOJ    ]}
{P    }  {      [,RESTART]}
         {partition       }
         {lineaddr[,EOJ]  }
```

uraddr Specifies the unit record device address associated with the reader/writer task being stopped:

cuu or X'cuu'

EOJ Command to stop the task when it has completed processing the current job entry. This parameter should always be specified for RJE tasks, unless an emergency occurs.

RESTART Specifies that when the task is started again, processing of the stopped job entry will begin at the record following the last one processed before the PSTOP command was issued. RESTART applies only to output processing.

Note: If RESTART is not specified along with uraddr, processing will begin at the first record of the current job entry when this task is started again (via PSTART).

partition

Specifies the POWER/VS controlled partition to be stopped as:

BG, F4, F3, F2, or F1

When a partition is stopped by a PSTOP command it is also released by POWER/VS.

This form of the PSTOP command is not effective until processing of the current reader entry is completed.

After the partition is released by POWER/VS, SYSRDR and all assignments to POWER/VS controlled devices for this partition are released. (It is your responsibility to reassign these devices when the partition is restarted.) The partition can now be started as s normal batch partition. To restart it as a POWER/VS controlled partition,the DOS/VS STOP or UNBATCH command should be issued, followed by a PSTART command.

lineaddr

Specifies the line address associated with the RJE task. It has the format:

cuu or X'cuu'

Examples:

PSTOP 00E,EOJ

Stop writer task operating on 00E after completion of the current list entry.

PSTOP F3

Release F3 from control of POWER/VS after processing of the current reader entry is completed.

PSTOP 00D,RESTART
: Stop writer task operating on 00D. When the task is restarted, processing of the stopped entry continues at the point of interruption.

PSTOP 030
: Stop the RJE task operating on line 030, immediately. If this task is reading, the input records already processed are lost. If the task is writing, output starts from the first record of the list entry when the task is restarted.

POWER/VS RJE Terminal Commands

Remote job entry (RJE) is an extension of POWER/VS and allows the user to submit jobs from remote locations via IBM 2770, 2780, 3741, or 3780 terminals for processing at a central installation under DOS/VS.

The central installation and the terminals can communicate through messages and commands. POWER/VS broadcast messages (no ALLUSERS-type) and responses to POWER/VS terminal commands are sent to the remote operator in the format of list output. They are stored in virtual storage and automatically printed (and subsequently deleted) whenever the line is available for print operations.

ALLUSERS-type broadcast messages are stored in virtual storage. They are displayed only upon request.

If an I/O error occurs, RJE will write SYSREC records to the recorder file.

FUNCTION OF POWER/VS RJE TERMINAL COMMANDS

POWER/VS RJE commands are used to control job entries that are entered at a terminal for processing at a central installation. There are four types of commands:

Terminal control commands are used to start and stop user sessions.

Task management commands are only applicable to an RJE writer task. The RJE reader task is started by the central operator when he brings up the line, and its operation is controlled by the system as are execution processors.

Queue management commands are only applicable to jobs that are submitted by or routed to the same remote-id as the one issuing the command.

Miscellaneous commands print the page layouts or transmit messages.

Format of POWER/VS RJE Terminal Commands

POWER/VS RJE terminal commands are entered on punched cards through the reader at the terminal. They must be submitted outside POWER/VS job boundaries, otherwise they are treated as user data.

Each RJE command consists of the following fields:

1 Identification field: Contains * in column 1, blank in column 2, and .. (periods) in columns 3 and 4. Column 5 must be blank.

2 Operation field: Specifies the RJE command. Must be preceded by one or more blanks in column 5. At least one blank must separate the operation and operand field.

3 Operand field: Contains one or more parameters, separated by commas. At least one blank must separate the operand field and the comments field.

4 Comments field: May contain any information considered helpful by the user. Continuation of the comments field is not allowed.

The operation field, operand field, and comments field must be contained within columns 6 through 71. Column 72 must be blank.

5 Sequence field: Sequence numbers are useful when a number of commands with the same operation code are submitted from a terminal. Columns 79 and 80 are returned in diagnostic messages.

Description of POWER/VS RJE Terminal Commands

Figure 10 shows all valid terminal commands.

The following pages describe all RJE terminal commands in alphabetical order. A summary of these commands is contained in Appendix C.

Type	Command	Operation
Terminal control	SIGNON SIGNOFF	Start a user session Terminate a user session
Task management	START STOP FLUSH RESTART GO	Start a writer or resume message receipt Stop a writer or stop message receipt Flush an active job Restart a writer entry Reactivate a task waiting
Queue management	DISPLAY ALTER DELETE RELEASE	Display job status Alter job attributes Delete a job or a message Release a job
Miscellaneous	BRDCST SETUP	Transmit a message Print page layout

Figure 10. POWER/VS RJE Terminal Commands

ALTER Command

The ALTER command is issued to change the various operands of a job entry after it has been logged in POWER/VS. The operands are the same as those specified in the * $$ JOB, * $$ LST, and * $$ PUN statements and have the same format. Any number of operands can be modified by one command. However, the remote user can alter attributes of only those jobs that were submitted by himself or were routed to him.

If an input queue entry is specified and an execution processor is currently processing this entry, message 1R88I is issued.

The priority of an entry in the reader queue is identical to that of the corresponding entries in the writer queues. On the other hand, disposition and class may be totally different in corresponding reader and list/punch queue entries, since input disposition and class are defined in the * $$ JOB statement, whereas output disposition and class are defined in the * $$ LST and * $$ PUN statements.

The COPY and REMOTE operands apply to the output only. When applied to a reader entry, they are ignored. If a writer entry is specified in the ALTER command and a writer task is currently processing the entry, only the number-of-copies can be altered. If any other parameter is specified, message 1R88I is issued and the command is ignored. The same message is issued if the output is segmented and the first segment is already deleted from the output queue.

No confirmative message is issued after the ALTER command. DISPLAY can be used for verification.

Format

```
                   ┌jobname[,jobnumber]┐
* .. ALTER queue  ─┤ALL                ├─
                   │*abc               │
                   └class1             ┘

                [,PRI=priority]
                [,DISP=disposition]
                [,CLASS=class2]
                [,COPY=number-of-copies]
                [,REMOTE=remid]
```

queue — Specifies the queue for which the command is to be executed:

LST = list queue
PUN = punch queue
RDR = read queue

jobname — Specifies the name by which the queue entry is known to POWER/VS. jobname can be two to eight alphameric characters.

jobnumber — Specifies the number assigned to the queue entry by POWER/VS. The jobnumber may be one to five digits long. The DISPLAY command may be used to obtain the jobnumber.

ALL — All job entries in the specified queue are to be altered.

*abc — All job entries with the same first characters are to be altered. abc represents any combination of up to seven alphameric characters.

class1 — All job entries of a certain class are to be altered in the specified queue.

priority Specifies the priority of the job entry. Priority is specified as a single digit from 0 to 9. Nine is the highest priority.

disposition
Can be specified as H for hold, K for keep after processing, L for leave, or D for delete after processing.

class2 Can be specified as any alphabetic character from A through Z. Also, specifications from 0 through 4 are allowed for input classes.

remid Can be specified as any number from 0 to 200. 0 indicates the central location.

Only the TO remote-id can be modified with an ALTER command; the FROM remote-id cannot be modified.

number-of-copies
Specifies the number of copies to be produced after the command is issued, including the one in progress.

Example:

```
* .. ALTER LST,PAYROLL,DISP=D,CLASS=B
```
Alter the disposition of the output of job PAYROLL to D, and alter the class to B.

BRDCST Command

The BRDCST command enables the remote operator to communicate with the central operator and other remote users. Messages can be submitted for one specific user or for all users.

Messages for specific remote-id's are displayed at once (if the terminal is ready to receive the message).

ALLUSERS-type messages are queued in virtual storage and displayed at the user's request only (DISPLAY message).

Format

```
* .. BRDCST   remid,'text'
```

remid Specifies the remote-id to which the command applies. Remote users can be identified by numbers from 1 to 200. A specification of 0 indicates the central location, and a specification of ALLUSERS indicates all users.

'text Specifies the text of the message. The message can consist of from 1 to 59 characters enclosed in single quotation marks. A single quotation mark within the message must be written as two quotation marks.

Examples:

```
* .. BRDCST 150,'Payroll will be
completed in 30 min.'
```
Broadcast the message to the user whose remote-id is 150.

```
* .. BRDCST 0,'Inventory is finished'
```
Broadcast the message to the central operator.

DELETE Command

The DELETE command is used to remove one or more job entries from the specified queue. Job entries being prosessed by an execution processor or write task are not affected by this command. The remote user can delete only jobs or messages that were submitted by himself or were routed to him.

Format

```
* .. DELETE  ⎧queue,jobname[jobnumber]⎫
             ⎪queue,ALL               ⎪
             ⎨queue,class             ⎬
             ⎪queue,*abc              ⎪
             ⎩MSG[,n]                 ⎭
```

queue Specifies the queue for which the command is to be executed:

LST = list queue
PUN = punch queue
RDR = read queue

jobname Specifies the name by which the job entry is known to POWER/VS. Jobname can be two to eight alphameric characters.

jobnumber
Specifies the number assigned to the job entry POWER/VS. The jobnumber is from one to five digits long. The DISPLAY command can be used to obtain the jobnumber.

ALL All job entries are to be deleted from the specified queue.

class All job entries of a certain class are to be deleted from the specified queue.

*abc All job entries with the same first n characters are to be deleted. abc represents any combination of up to seven alphameric characters.

MSG[,n] ALLUSERS-type message is to be deleted, where n is the message number. The message number can be obtained with a DISPLAY MSG command. If n is omitted, all ALLUSERS-type messages that were entered from this remote-id are deleted. ALLUSERS-type messages can only be deleted by their originator or by the central operator.

Examples:

```
* .. DELETE RDR,ALL
        Delete the entire reader queue.

* .. DELETE LST,B
        Delete all entries in the list
        queue with a class of B.

* .. DELETE MSG
        Delete all ALLUSERS-type messages
        entered by this remote-id.
```

DISPLAY Command

he DISPLAY command provides the remote perator with a status report of the specified job entry or entries. The remote operator can only display the status of jobs that he submitted himself or that have been routed to him.

Each job entry is represented by one line containing the following items:

Items	Characters
jobname	8
jobnumber	5
priority	1
disposition	1
class	1
FROM remote-id	3
TO remote-id	3
number-of-records/pages	6
number-of-copies	2
forms-number	4

The last four items are displayed for output only. For punch output, the number of records is displayed, whereas for list output the number of pages is displayed.

Each line of the status report starts with the message number 1R46I. If the status report is made up of entries from ifferent queues, the queue name is printed efore the first job entry is displayed.

The disposition of a job entry may appear as H for hold, K for keep after processing, L for leave, D for delete after processing, or as an * if an execution processor or a write task is currently processing the entry.

The class can appear as an alphameric character from A to Z, or from 0 to 4. The remote-id is displayed as a number from 001 to 200 if the job is entered from a terminal or if the output is routed to a terminal. The remote-id of the central location (000) is not printed in the status report.

If a write task is processing an entry when the DISPLAY command is entered, number-of-copies is the number of copies left to be produced, including the one in progress. The number-of-records/pages is the number of records or pages left to be produced. The operator can use this information to determine whether to stop a write task or let it continue until the current entry is completed.

Output from the DISPLAY command is stored in virtual storage, and is transmitted as soon as the line becomes available for write operations.

Format

```
* .. DISPLAY  / queue,jobname[,jobnumber] \
              | queue[,ALL]               |
              | queue,HOLD                |
              | queue,FREE                |
              | queue,*abc                |
             <  queue,class                >
              | ALL                       |
              | HOLD                      |
              | FREE                      |
              | *abc                      |
              \ MSG                       /
```

queue Specifies the queue for which the DISPLAY command is to be executed:

LST = list queue
PUN = punch queue
RDR = read queue

jobname Specifies the name by which the job entry is known to POWER/VS. The jobname can be two to eight alphameric characters.

jobnumber
Specifies the number assigned to the job entry by POWER/VS. The jobnumber is one to five digits long.

ALL Requests the status of all entries in the specified queue. If the queue parameter is not specified, status information on all entries in all queues is displayed.

HOLD Requests the status of all entries in the specified queue that are not available for processing (that is, in hold state or leave state). If the queue parameter is not specified, status information is displayed on all such entries.

FREE Requests the status of all entries in the specified queue that are available for processing (that is, in keep-after-processing state or delete-after-processing state). If the queue parameter is not specified, status information is displayed on all such entries.

class Requests the status of all job entries in the queue with the specified class. The class can be specified as an alphameric character from A through Z, or from 0 to 4.

*abc Requests the status of all entries having the first characters of their jobnames in common. abc represents any combination of from one to seven characters. If the queue parameter is not specified, status information is displayed for all queues.

MSG Requests the display of all ALLUSERS-type messages that have been entered, together with their originators.

```
    d all  DISPLAY THE STATUS OF ALL ENTRIES IN ALL QUEUES
F1  1R46I  READER QUEUE   P D C FROM
F1  1R46I  VERA02    00003 3 D 4 022
F1  1R46I  VERA11    00012 3 D 4 004
F1  1R46I  VERA01    00002 3 K B 015
F1  1R46I  VERA03    00004 3 D B 012
F1  1R46I  VERA04    00005 3 D B 005
F1  1R46I  VERA05    00006 3 D B 004
F1  1R46I  VERA06    00007 3 D B 000
F1  1R46I  VERA07    00008 3 D B 000
F1  1R46I  VERA08    00009 3 * P 003
F1  1R46I  VERA09    00010 3 D P 011
F1  1R46I  VERA13    00014 3 D P 009
F1  1R46I  VERA10    00011 2 D P 006
F1  1R46I  AUTONAME 00013 1 D Q 010
F1  1R46I    LIST QUEUE   P D C FROM TC  PAGES CC FORM
F1  1R46I  VERA08    00009 3 D X 003 000      4  1
F1  1R46I   PUNCH QUEUE NOTHING TO DISPLAY
```

FLUSH Command

The FLUSH command is used to discontinue the job entry currently being processed by an RJE writer task.

When the writer task is flushed, further output processing is terminated for this POWER/VS job, and the writer entry is deleted, unless HOLD is specified.

FLUSH is invalid when applied to an inactive task. Also, FLUSH can only be issued for a task that has been interrupted by the remote or central operator or by an action-type POWER/VS message.

<u>Format</u>

```
* .. FLUSH {task      }
           {task,HOLD}
```

task Specifies the task to be deleted as IST or PUN.

HOLD Specifies that the corresponding writer entry is not to be deleted, but put in the hold state.

<u>Examples</u>:

```
* .. FLUSH PUN
     Delete the punch entry
     currently being punched.
     Continue with the next
     entry.

* .. FLUSH LST,HOLD
     Skip the list entry
     currently being printed, but
     do not delete it from the
     queue. Continue with the
     next entry.
```

GO Command

The GO command is used to reactivate a writer task after receiving the "mount forms" message. (Message number 1Q40A.)

<u>Format</u>

```
* .. GO task
```

task Specifies the task to be reactivated as LST or PUN.

<u>Example</u>:

```
* .. GO LST
     Reactivate the list task.
     The proper forms have been
     mounted.
```

RELEASE Command

The RELEASE command is used to take one or more job entries out of the hold or leave state and make them available for processing. After processing, a job entry originally in the leave state returns to the leave state; a job in the hold state is deleted from the queue.

If a RELEASE command is issued for a job entry that is not in the hold state or leave state, a message is issued and the command is ignored.

The remote user can release only those jobs that were submitted by himself or were routed to him.

<u>Formats</u>

```
* .. RELEASE {queue,jobname[,jobnumber]}
             {queue[,ALL]              }
             {queue,class              }
             {queue,*abc               }
```

queue Specifies the queue for which the command is to be executed:

LST = list queue
PUN = punch queue
RDR = read queue

jobname Specifies the name by which the job entry is known to POWER/VS. The jobname can be two to eight alphameric characters.

jobnumber
Specifies the number assigned to the job entry by POWER/VS. The jobnumber is one to five digits long. The DISPLAY command can be used to obtain the number.

ALL All job entries are to be released from the specified queue.

class All job entries of a certain class are to be released from the specified queue.

*abc All job entries with the same first characters are to be released. abc represents any combination of up to seven alphameric characters.

Examples:

```
* .. RELEASE LST,PAYROLL
     Release the entry PAYROLL
     from list queue.

* .. RELEASE RDR,ALL
     Release the entire read
     queue.

* .. RELEASE LST,*PAY
     Release all entries in the
     list queue that begin with
     PAY.
```

RESTART Command

The RESTART command directs the RJE writer task to discontinue printing or punching immediately and to resume output from a specified point. A list writer can resume printing from either the beginning of the list entry, or up to 9999 pages forward or backward from the interrupted page. A punch writer can resume punching from either the beginning of the punch entry, or up to 9999 cards forward or backward.

The number of duplicated pages printed or cards punched because of a RESTART command is separately counted and stored in the list account record (if created).

RESTART is invalid when applied to an inactive task. Also, RESTART is only applicable to a task that has been interrupted by the operator or by an action-type POWER/VS message.

Format

```
* .. RESTART {task  }
             {task,n}
```

task — Specifies the task to be restarted as LST or PUN.

n — Specifies a signed or unsigned value from 0 to 9999. A plus sign indicates the number of pages or cards forward and a minus sign indicates the number of pages or cards backward from the last interrupted page or card. No sign indicates a specified page or card count from the beginning.

If the operand is omitted or is too large a backward count, output resumes from the first page or card. If too large a forward count is specified, message 1Q42I is issued and the command is ignored.

Example:

```
* .. RESTART LST,-10
     Reprint the previous ten
     pages and continue to the
     end of output.
```

SETUP Command

The SETUP command can be entered only after message 1Q40A requesting a special forms mount is displayed. It causes n pages of spooled list data to be printed. All printable characters of the page are replaced by the character X. During the printing of the n pages, the operator can manually align the printer forms.

The number of pages printed by SETUP are counted and stored separately in the list account record.

When the GO command is issued to resume printing, the pages are printed again with their original contents.

Format

```
* .. SETUP LST[,n]
```

LST,n — One or two digits that specify the number of pages to be printed. If n is omitted, only one page is printed.

Example:

```
* .. SETUP LST,2
     Print two pages to allow forms
     alignment.
```

```
                                                                  xxxx xxx   xx

      xxx xx xxx xxxxx xxxxxxxxxxxxxxx xxxxxxxx                     xxxxxxxxx
1st
Page  xxx xxxxxxx xxxxxxxxxxxx xxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxx xxxxxxx

           x xxxx



                                                                  xxxx xxx   xx

     xxx xx xxx xxxxx xxxxxxxxxxxxxxxxxxx xxxxxxxx                  xxxxxxxxx
2nd
Page    xxxx xxxxxxx xxxxxxxxxxx xxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxx xxxxxxx

           x xxxx



                                                                    SEPT 30,   74

      PAY TO THE ORDER OF..........ONE EMPLOYEE                       ***123.45
1ST
Page     ONE HUNDRED TWENTY-THREE and ------------------------------ 45/100 DOLLARS

            * 2001



                                                                    SEPT 30,   74

      PAY TO THE ORDER OF..........ANOTHER EMPLOYEE                   ***543.21
2nd
Page     FIVE HUNDRED FORTY-THREE AND ------------------------------ 21/100 DOLLARS

             # 2002
```

SIGNOFF Command

The SIGNOFF command informs the system that the remote operator will terminate the session at the next end-of-file condition on the reader. POWER/VS will disconnect the terminal from the system, stop any active writers, and prepare the line for other terminals to sign on.

Note: A terminal is forced to sign off when it has been idle for a number of minutes, as defined during POWER/VS generation.

Format

```
* .. SIGNOFF    (no operand)
```

Example:

```
* .. SIGNOFF   Terminate a session.
```

SIGNON Command

The SIGNON command identifies the terminal, and makes the logical connection between the terminal and the system. It must be the first card submitted from the terminal after the line is started by the central operator. It can also be submitted during the course of the session, in which case it terminates the current session and initiates the next session.

The operator must use the remote-id assigned to him during POWER/VS generation, since the terminal characteristics are defined in relation to this remote-id. If the same remote-id is used to sign onto another terminal with different hardware characteristics, the results are unpredictable.

The security password must match the password set for the line by the central operator. If the central operator did not specify a password when the line was started, it must match the password set for the line during POWER/VS genêration. If no password is set for the line, the password parameter can be omitted.

Format

```
* .. SIGNON remid[,password]
     [,user-information]
```

remid Specifies the remote-id assigned to the terminal during POWER/VS generation. Remote users are identified by numbers from 1 to 200.

password
Specifies the password set for the line by the central operator, or set during POWER/VS generation. It can be any combination of up to eight alphameric characters.

user-information
Consists of up to sixteen bytes of user information, which are moved to the line account record. This information is punched in columns 56 to 71.

Note: An invalid SIGNON command is interpreted as a SIGNOFF command. The terminal is disconnected if it is on a switched line.

Example:

```
* .. SIGNON    002,SECRET
```

Sign on at a terminal with remote-id 002 and password SECRET.

START Command

The START command is used to initiate an RJE writer task (writer). The RJE reader task (reader), is controlled only by the central operator.

The remote operator must enter a START command in order to receive his output. POWER/VS does not send unsolicited output to any terminal.

A START command is invalid when applied to an active task. The START command is also used to resume message receipt (broadcast and system messages). * .. START MSG is only meaningful if an * .. STOP MSG was given previously. (Refer to the STOP command.)

Format

```
* .. START {task      }
           {task,class}
           {MSG       }
```

task Specifies the task to be started as LST or PUN. Writer entries are processed on a first-in, first-out basis within priority, within class.

class Defines the output class(es) upon which the task is to operate. Up to four classes can be designated by specifying one to four alphabetic characters from A through Z. The order specified is the order in which the classes will be processed. If no class parameter is specified, only class A is selected.

MSG Specifies that all subsequent messages directed to this terminal are to be accepted (printed).

Examples:

```
* .. START PUN
```

Start punch task for class A.

```
* .. START LST,AB
```

Start list task for classes A and B.

STOP Command

The STOP command is used to terminate an RJE writer task. The terminal operator cannot control the operation of an RJE reader task.

A STOP command is invalid when applied to an inactive task. The STOP command is also used to cause broadcast messages to be ignored (not accepted). After STOP MSG, all messages (broadcast and system) for this user are discarded.

Format

```
* .. STOP {task        }
          {task,EOJ    }
          {task,RESTART}
          {MSG         }
```

task Specifies the task to be stopped. LST or PUN are the valid entries.

EOJ Specifies that the task will not stop until it has completed processing the current entry.

RESTART Specifies that when the task is started again, processing will begin at the record following the last one processed before the STOP command was issued.

MSG Specifies that all subsequent messages directed to this terminal are to be ignored. In addition, existing messages in the queue are deleted immediately.

Note: If either EOJ or RESTART is not specified along with task, processing will begin at the first record of the current job entry when this task is started again (via START).

Examples:

```
* .. STOP PUN,RESTART
        Stop the punch task.  When the
        task is restarted, processing of
        the entry continues at the point
        of interruption.

* .. STOP LST,EOJ
        Stop the list task after
        completion of the current list
        entry.

* .. STOP LST
        Stop the list task immediately.
        When the task is restarted,
        processing will start at the
        beginning of the interrupted list
        entry.

* .. STOP MSG
        Ignore messages.
```

Linkage Editor

The linkage editor prepares programs for execution with DOS/VS and accepts as input the relocatable object modules produced by the language translators and object modules produced by the CSERV programs. It processes these modules into program phases, which may be immediately executed or cataloged into the core image library.

Job control calls the linkage editor program when a // EXEC LNKEDT control statement is read. Control is returned to job control when the link edit function is completed, unless the CATAL option was specified, in which case the CATAL function is completed before control is returned to job control.

The linkage editor program can run in either the background or a foreground partition. If it runs in the foreground, you must assign a private core image library (SYSCLB). In the background partition, the linkage editor defaults to the system core image library if nc private core image library is assigned. The linkage editor issues a diagnostic message and terminates abnormally when you assign the private core image library across partitions.

If the supervisor supports the relocating loader, you do not need to write a self-relocating program to enable that program to execute in any real or virtual partition. The linkage editor will produce relocatable phases whenever possible. The linkage editor determines whether a phase can be made relocatable by inspecting the origin of the PHASE statement.

The linkage editor control cards direct the program to read an input module(s) and to form phases from the control sections within the modules. The linkage editor relocates the origin of each control section in the phase, assigns each phase an area of storage and a transfer address, and modifies the contents of the address constants in the phase.

The relocation factor for each control section is determined and saved by building a table called the control dictionary. This table contains the linkage editor phase definitions and the module ESD items. When complete, the table provides sufficient information for determining the location of each control section and for resolving any references between control sections.

The module TXT items are then built into phase blocks. The RLD items (address constants) are modified and inserted into the text. A transfer address is determined for each phase.

Language Translator Modules

The input to the linkage editor consists of object modules and linkage editor control cards. Each module is the output of a complete language translator run. It consists of dictionaries and text for one or more control sections.

The dictionaries contain the information necessary for the linkage editor to resolve references between different modules. The text consists of the actual instructions and data fields of the module.

Six card types can be produced, by the language translators or by the proarammer, to form a module. They appear in the following order:

Card Type	Definition
ESD	External symbol dictionary
SYM	Ignored by linkage editor
TXT	Text
RLD	Relocation list dictionary
REP	Replacement to text made by the programmer
END	End of module.

For the format of each of these cards (except SYM), see "Appendix B: Linkage Editor Summary".

The external symbol dictionary contains control section definitions and intermodule references. When the linkage editor has the ESDs from all modules, it can relocate the sections and resolve the references. Six types of entries are defined in the control dictionary.

ESD Type	Definition
SD	Section definition: provides control section name, assembled origin and length. Generated by a named START or a named CSECT in a source module.
WX	Generated by Weak External Reference (WXTRN), which has a function similar to EXTERN, except that WXTRN suppresses AUTOLINK. The linkage editor treats WX as an ER, NOAUTO.
PC	Private code: provides assembled origin and length for an unnamed control section.
LD/LR	Label definition: specifies the assembled address and the associated SD of a label that may be referred to by another module. The LD entry is termed LR (Label Reference) when the entry is matched to an ER entry.
ER	External reference: specifies the location of a reference made to another module. ER is generated by EXTRN or a V-type address constant in a source module.
CM	Common: indicates the amount of storage to be reserved for common use by different phases. CM is generated by COM in a source module.

The relocation list dictionary identifies portions of text that must be modified on relocation (address constants).

When the linkage editor reads a module, it stores ESDs in its control dictionary, writes TXT and REP items in core image blocks in the library, and writes RLD items on an RLD file. Each item that is identified by the language translators with an ESID number is reidentified by the linkage editor with a control dictionary number to avoid duplication between modules. All programs to be executed under DOS/VS must be link-edited and stored in the core image library. The core image library is either on SYSRES (the system core image library) or on SYSCLB (a private core image library). The linkage editor program operates in one of three modes:

- Catalog mode. An object module is link-edited and permanently stored in the core image library. The core image directory for cataloged phases is updated. If the phase is eligible for the shared virtual area and is indicated as SVA-eligible in the system directory list, the phase is also loaded into the SVA.
- Load and execute mode. An object module is link-edited for temporary storage in the core image library and is immediately executed.
- Assemble and execute mode. A source module is assembled or compiled. The object module (output) is link-edited for temporary storage in the core image library and is immediately executed.

Linkage Editor Control Statements

In addition to the program cards previously listed, object modules used as input for the linkage editor include linkage editor control statements. There are four kinds of these control statements, each of which is described in detail further on in this section.

PHASE   Indicates the beginning of a phase. It gives the name of the phase and the storage address where it is to be loaded.

INCLUDE   Signals that an object module or parts thereof are to be included.

ENTRY   Provides an optional transfer address for the first phase.

ACTION   Specifies options to be taken.

General Control Statement Format

The linkage editor control statements are similar in format to statements processed by the assembler. The operation field must be preceded by one or more blanks. The operation field must begin to the right of column 1 and must be separated from the operand field by at least one blank position. The operand field is terminated by the first blank position. It cannot extend past column 71.

Control Statement Placement

If multiple-object modules are being prepared in a single linkage editor run, the single ENTRY statement should follow the last object module. The ACTION statement(s) must be the first record(s) encountered in the input stream; otherwise, they are ignored.

PHASE and INCLUDE statements may be present on SYSRDR, SYSIPT, or in the relocatable library. Figure 11 shows the possible placement of the PHASE and INCLUDE statements.

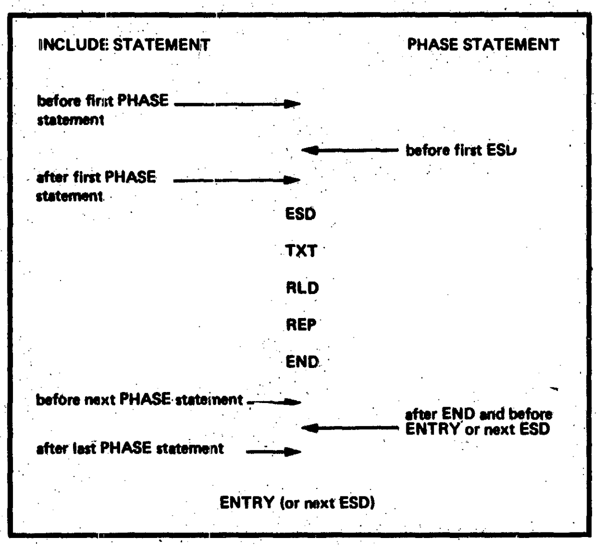

INCLUDE STATEMENT
PHASE STATEMENT
before first PHASE statement
before first ESD
after first PHASE statement
ESD
TXT
RLD
REP
END
before next PHASE statement
after END and before ENTRY or next ESD
after last PHASE statement
ENTRY (or next ESD)

Figure 11. Placement of PHASE and INCLUDE Statements

INCLUDE statements within modules in the relocatable library must precede the ESD statement for the module.

PHASE Statement

The PHASE statement must precede the first object module of each phase processed by the linkage editor. Under no circumstances can a PHASE statement occur within a control section. There can be several control sections within a phase. When several PHASE statements appear before an object module, each of the statements must be followed by at least one INCLUDE statement. Any object module not preceded by a PHASE statement is included in the current phase.

The first (or only) object module input for the linkage editor should include a PHASE control statement before the first ESD item. If no PHASE statement is used, or if the PHASE statement is in error, the linkage editor constructs a dummy statement. This allows testing of the program when the LINK option is used. However, the program with the dummy PHASE statement cannot be cataloged in a core image library; when the CATAL option is used, the job is canceled. The last (or only) object module may optionally be followed by an ENTRY control statement.

This statement provides the linkage editor with a phase name and an origin point for the phase. The phase name is used to catalog the phase into a core image library. This name is used in a FETCH or LOAD macro to retrieve the phase for execution. The PHASE statement is in the following format.

```
PHASE  name,origin[,NOAUTO][,SVA][,PBDY]
```

At least one blank must precede PHASE. The operands have the following meaning:

name — Symbolic name of the phase. One to eight alphameric (0-9, A-Z, #, $, /, and @) characters are used as the phase name. The name may not be ALL.

Each single-phase program should be unique in the first four characters of its phase name. Any phases that are named by the same first four characters will be classified as a multiphase program. When fetching a phase of a multiphase program, the partition must be large enough to contain the largest phase.

origin — Specifies the load address of the phase. If the origin of a phase was specified relative to the beginning of a partition, the linkage editor refers to the start address of the virtual partition to calculate the load address of the phase.

To run a program in a real partition, you can

- Link-edit the program with ACTION REL (see the ACTION statement) so that it can be relocated to a real partition when it is loaded.

- Write the program to be self-relocating.

- Link-edit the program with a PHASE statement that contains the absolute address of the location within the real partition where the program is to be loaded.

If COMMON is used, the length of the largest COMMON is added to every phase origin, even if the origin is given as an absolute value. COMMON is located at the beginning of the phase with the lowest origin address (if multiple phases).

The load address can be in one of six forms:

1. symbol[(phase)] [±relocation]
2. *[±relocation]
3. S[±relocation]
4. ROOT

} relative addresses

5. +displacement
6. F+address

} absolute addresses

A phase can be made relocatable if its origin is specified as a relative address (formats 1-4 above). However, if the address is relative to another phase which is not relocatable, the new phase will not be relocatable. Refer to the ACTION statement for additional information about the relocating loader.

The elements that make up the six forms that specify the origin signify the following.

1. symbol: May be a previously defined phase name, control section name, or external label (the operand of an ENTRY source statement).

   (phase): If symbol is a previously defined control section name or a previously defined external label that appears in more than one phase, phase (in parentheses) directs the linkage editor to the phase that contains the origin. The phase name must have been defined previously.

   relocation: Indicates that the origin of the phase currently being processed will be set relative to the symbol by a relocation term consisting of a + or a - immediately followed by: X'hhhhhh' (one to six hexadecimal digits), dddddddd (one to eight decimal digits), or nK.

2. *: The linkage editor assigns the next storage location in the virtual partition (with forced doubleword alignment) as an origin for the next phase.

   For the first PHASE statement in the background partition, * indicates that the origin is to be the first doubleword storage address after the partition save area, the label save area (if any), and the area assigned to the COMMON pool (if any).

   relocation: Indicates relocation of the phase relative to the next storage location of the virtual partition. The format is as specified in item 1.

3. S: If S is specified, the origin is determined in the same manner as the first PHASE statement in item 2.

   relocation: Indicates relocation of the phase relative to the start of the virtual partition as described in item 2.

4. ROOT: Tells the linkage editor that the phase that follows is a root phase. The storage address assigned to the root phase is determined in the same manner as the first PHASE statement in item 2. Only the first PHASE statement is permitted to specify ROOT. Any qualitative information (phase or relocation) is ignored when ROOT is specified. If a control section (CSECT) appears in the root phase, other occurrences of the same control section are ignored and all references are resolved to the control section in the root. (This does not apply to control sections and external references that begin with the letters IBM.) Control sections are not duplicated within the same phase. If any subsequent phase overlays any part of the ROOT phase, a warning diagnostic is displayed on SYSLST if ACTION MAP is specified. Refer also to ACTION Statement.

5. +displacement: Allows the origin (loading address) to be set at a specified location. The origin point is an absolute address, relative to zero.

   displacement must be: X'hhhhhh' (one to six hexadecimal digits), dddddddd (one to eight decimal digits), or nK. A displacement of zero (+0) would be used to denote a self-relocating program.

6. F+address: This format allows the origin of the program that is being link-edited in one partition to be set at the start of another partition that is not allocated. If the other

partition is allocated, ACTION BG or ACTION Fn has the same effect as F+address.

It also indicates that an area is to be reserved at the beginning of the other partition for the program name, a register save area, and label information. F should never be used for self-relocating programs. If COMMON is used, the COMMON start address is resolved to the first doubleword boundary after the reserved area at the beginning of the area specified by the F+displacement in the PHASE card.

address: The absolute storage address of the partition in which the link-edited program is to be executed. It may be specified by: X'hhhhhh' (four to six hexadecimal digits), dddddddd (five to eight decimal digits), or nnnnK (n is two to four digits). For example, an address may be specified as +32K or +X'8000' or +32768. The origin of the phase is on the first doubleword boundary after the sum of address, the adjustment for the save area requirements, the label area and the length of the COMMON area, if applicable.

NOAUTO Indicates that the Automatic Library Lookup (AUTOLINK) feature is suppressed for both the private and system relocatable libraries. AUTOLINK collects each unresolved external reference from the phase. It then searches the private relocatable library (if assigned) and then the system relocatable library for a cataloged object module with the same name as each unresolved external reference. When a match is found, the module in the private or system relocatable library is link-edited into the phase. The AUTOLINK retrieved module must have an entry point matching the external reference in order to resolve its address. Unresolved external references are processed sequentially in alphameric order. Object-module cross references with labels identical to library object-module entry-point labels are erroneous. The use of the NOAUTO operand in a PHASE statement causes the AUTOLINK process to be suppressed for that phase only. (Also see ACTION Statement.)

SVA Indicates that the phase is SVA-eligible. This means that the phase must be reenterable and relocatable (link-edited for loading by the relocating loader). When this phase is cataloged into the system core image library, the linkage editor will also have the phase loaded into the SVA if the phasename was listed in the SDL with an SVA operand. If the linkage editor finds that a phase that is specified with the SVA operand is not relocatable, an error message is issued and the SVA operand is ignored.

PBDY Indicates that the phase is to be link-edited on a page boundary. If the current link-edit address is not aligned on a page boundary, the linkage editor uses the next higher page boundary address.

Note: It is not recommended that PBDY be specified for the first or only phase of a program.

Some examples of PHASE statements follow.

```
PHASE PHNAME,*+504
```

This causes loading to start 504 bytes past the end of the previous phase.

```
PHASE PHNAME3,PHNAME2
```

This causes loading to start at the same point where the loading of phase PHNAME2 started.

```
PHASE PHNAME,ROOT
```

Loading begins at the first doubleword after the beginning of the partition, the partition save area, the label save area (if any), and the area assigned to the COMMON pool (if any). When the PHASE statement contains a ROOT origin, this PHASE statement must be the first PHASE statement read by the linkage editor. Otherwise, it is treated as a symbol.

```
PHASE PHNAME,CSECT1(PHNAME2)
```

This causes loading to start at the point where CSECT1 was loaded. CSECT1, the named control section, must have appeared in the phase named PHNAME2.

```
PHASE PHNAME,F+X'6000'
```

This causes loading to start at 24K plus the length of the save area and label area.

```
PHASE PHNAME,F+32K
```

This causes loading to start at 32K plus the length of the save area and label area.

```
PHASE PHNAME1,F+30K
PHASE PHNAME2,*
PHASE PHNAME3,PHNAME2
```

The first phase (PHNAME1) of the preceding series is loaded starting at 30K plus the length of the save area and label area. The second phase (PHNAME2) of the series is loaded at the end of PHNAME1. The third phase (PHNAME3) is loaded at the same address as was PHNAME2, that is, at the end of PHNAME1.

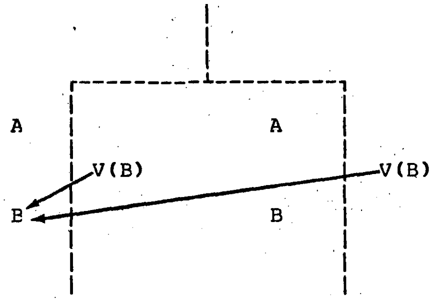
Note: In each of the preceding examples, if the origin address supplied is not on a doubleword boundary, the linkage editor automatically increments to the next doubleword boundary.

The linkage editor allows the inclusion of the same control section within each of several phases. If a control section (CSECT) appears in a ROOT phase, it does not appear in any other phase. (This does not apply to control sections that begin with the letters IBM.) A duplicate control section within the same phase will be ignored.

As external references occur in a phase, they are resolved preferentially with the entry point within the ROOT phase (if any), or the last previous occurrence of this entry point. For example, the coding

```
A    START
       .
       .
       .
     DC V(B)
       .
       .
       .
B    CSECT
       .
       .
       .
     END
```

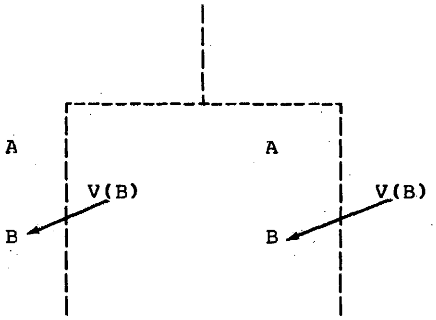
when used as a module in two phases has two different results. When the module is part of the ROOT phase, the external reference [DC V(B)] is resolved with the entry point within the ROOT phase (B CSECT). When the module is not part of the ROOT phase, the external reference is resolved with the last previous occurrence of this entry point. Since the reference is given before the CSECT labeled B is defined in that phase, it is resolved with the CSECT labeled B in the ROOT phase.



Whereas the coding

```
A    START
       .
B    CSECT
       .
A    CSECT
       .
       .
       .
     DC V(B)
       .
       .
       .
B    CSECT
       .
       .
       .
     END
```

when used as a module in two phases has the same result. When the module is part of the ROOT phase, the external reference [DC V(B)] is resolved with the entry point within the ROOT phase (B CSECT). When the module is not part of the ROOT phase, the external reference is resolved with the last previous occurrence of this entry point. Since the CSECT labeled B is defined in this phase before the reference is given, the external reference is resolved in the same phase. No problem arises from defining duplicate CSECTs in the same phase, since these are ignored.



This method of coding redefines the sequence of ESD information to allow valid cross reference by the linkage editor. This is also true in AUTOLINK mode, except for the case of privileged external references (external references whose labels begin with the letters IJ). For those references, if the resolution is not possible within the current phase or ROOT phase, then the AUTOLINK function is performed on this external reference at the end of the phase, and the other previously defined phases are not examined for possible resolution. If NOAUTO is specified, the IJ prefix is not privileged.

INCLUDE Statement

INCLUDE indicates that an object module is to be included for editing by the linkage editor. It has two optional operands. When both operands are used, they must be in the prescribed order. When the first operand is omitted and the second operand is used, a comma must precede the second operand. The first operand indicates that the input is in the relocatable library. The second operand indicates that the input is in submodular structure. The names appearing in the namelist (second operand) are the names of selected control sections from which a phase is to be constructed.

If both operands are omitted, the object module to be included is assumed to be on SYSIPT. Job control copies it onto SYSLNK. Each series of relocatable modules on SYSIPT must be terminated by a /* control statement. INCLUDE statements with blank operands are recognized only on SYSRDR.

If only the first operand is present, the object module is assumed to be in either the private or the system relocatable library. The linkage editor first searches the private relocatable library (if assigned) and then the system relocatable library for the module. The module name must be the same as that used when the module was cataloged in the library. Including modules from the relocatable libraries permits the programmer to include standard subroutines in his program at link-edit time.

If only the second operand is present, the object module to be included is assumed to be in the input stream (SYSLNK). The linkage editor reads the object module and extracts the control section(s) indicated by the second operand of the INCLUDE.

Note: If this option is elected, the module must be preceded by an INCLUDE statement with a blank operand in order for ,b control to place the module on SYSLNK.

If both operands are present, the object module is read from the relocatable library and the indicated control section(s) are extracted.

The placement of the INCLUDE statement determines the position of the module in the program phase. An included module (in the relocatable library) can be preceded by one or more additional INCLUDE statements.

The format of the INCLUDE statement is:

```
INCLUDE   [modulename][,(namelist)]
```

At least one blank must precede INCLUDE.

modulename
Symbolic name of the module, as used when cataloged in the relocatable library. It consists of one to eight alphameric characters.

(namelist)
The linkage editor constructs a phase from only the control sections specified. The namelist is in the following format.

(csname1,csname2,...)

Entries within the parentheses are the names of the control sections that will be used to constitute a phase. When the namelist option is used and only selected control sections are included in a phase, a submodular phase is created. The counterpart of a submodular phase is a normal phase. A normal phase contains all control sections of one or more object modules. It is possible to include within the same phase an object module(s) without the namelist option and an object module(s) specifying the namelist option. The total number of control sections in a namelist cannot exceed five; however, any number of INCLUDE statements can be used.

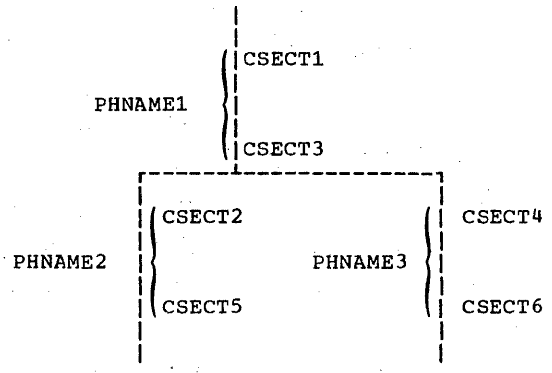
Modules in the relocatable library can be nested by using INCLUDE statements up to a depth of five (level of six). Modules included by INCLUDE statements read from SYSRDR are referred to as being in the first level. Modules included by statements in the first level are at the second level. Modules included by statements in the second level are at the third level, and so on up to six levels.

Submodular Structure

When several control sections are compiled together in one object module, it is sometimes desirable to break them up into several phases at link-edit time. This is done by using a PHASE statement followed by an INCLUDE statement with the namelist option. For example, in the sequence

```
PHASE    PHNAME1,*
INCLUDE  ,(CSECT1,CSECT3)
PHASE    PHNAME2,*
INCLUDE  ,(CSECT2,CSECT5)
PHASE    PHNAME3,PHNAME2
INCLUDE  ,(CSECT4,CSECT6)
```

the linkage editor structures the next module composed of CSECT1-CSECT6 in three overlays as shown·



The absence of the first operand in the INCLUDE statement indicates that the control sections are to be incorporated from the next succeeding module in the input stream.

The preceding sequence of PHASE and INCLUDE statements may be read by job control onto SYSLNK in one of two ways:

- If the PHASE and INCLUDE statements are on SYSRDR, an INCLUDE statement with a blank operand must follow the sequence to read the module (on SYSIPT) containing CSECT1-CSECT6 onto SYSLNK.
- If the PHASE and INCLUDE statements are on SYSIPT (immediately preceding the module), an INCLUDE statement with a blank operand on SYSRDR directs job control to read everything onto SYSLNK from SYSIPT down to the /* statement.

PHASE and INCLUDE statements can also be in the relocatable library. This implies that submodular phases can be constructed from modules in the relocatable library. If PHASE and INCLUDE statements come from the relocatable library (via an INCLUDE MODNAME), then the control sections for that module are in the relocatable library. In this structure, the required control sections (in the relocatable library) immediately follow the last INCLUDE statement. For example, in the sequence

```
PHASE    PHNAME1,*
INCLUDE  MODNAME1,(CSECT1,CSECT3)
PHASE    PHNAME2,*
INCLUDE  MODNAME1,(CSECT2,CSECT5)
PHASE    PHNAME3,PHNAME2
INCLUDE  MODNAME1,(CSECT4,CSECT6)
```

the linkage editor structures the next module (cataloged in the relocatable library under MODNAME1) composed of CSECT1-CSECT6 into the same three overlays as shown in the preceding example.

If MODNAME1 contains an INCLUDE statement, the linkage editor interprets this to mean that the module to be included should also be searched for the control sections requested in the namelist. For example, in the relocatable library if MODNAME1 contains

```
INCLUDE  MODNAME2
CSECT3
CSECT5
CSECT6
```

and in the relocatable library MODNAME2 contains

```
CSECT1
CSECT2
CSECT4
```

upon encountering an

```
INCLUDE  MODNAME1,(CSECT1,CSECT3)
```

statement, the linkage editor goes to MODNAME1 and finds INCLUDE MODNAME2. The linkage editor then goes to MODNAME2 and extracts CSECT1 and returns to MODNAME1 and extracts CSECT3.

A nonsubmodular INCLUDE statement may be placed before or after a submodular INCLUDE statement. This results in the addition of the included module into the phase at the point the INCLUDE statement is encountered. For example, if MOD1 contains CSECT4 and CSECT5, the sequence

```
PHASE    PHNAME1,*
INCLUDE  ,(CSECT1,CSECT3)
INCLUDE  MOD1
```

(Object module containing CSECT1 and CSECT3)

results in the following structure:

```
PHNAME1  CSECT1
         CSECT3
         CSECT4
         CSECT5
```

while the sequence

```
PHASE    PHNAME1,*
INCLUDE  MOD1
INCLUDE  ,(CSECT1,CSECT3)
```

(Object module containing CSECT1 and CSECT3)

results in the following structure:

```
PHNAME1  CSECT4
         CSECT5
         CSECT1
         CSECT3
```

Note: Both of the following statements produce the same result.

```
INCLUDE  ,(CSECT1,CSECT3)
INCLUDE  ,(CSECT3,CSECT1)
```

That is, CSECT1 and CSECT3 are in storage in that sequence. This is because the linkage editor extracts control sections in the order in which they appear in the input stream, not as they are ordered in the name list. In order to have CSECT3 physically located ahead of CSECT1 in storage, two INCLUDEs must be used:

```
INCLUDE  ,(CSECT3)
INCLUDE  ,(CSECT1)
```

As no diagnostic is given if a control section, specified in the namelist, is not present in the indicated module, you can inspect the MAP supplied by the linkage editor to determine if the proper control sections are in the correct phases.

ENTRY Statement

Every program, as input for the linkage editor, is terminated by an ENTRY statement. Its format is:

```
ENTRY       [entrypoint]
```

At least one blank must precede ENTRY.

entrypoint — Symbolic name of an entry point. It must be the name of a CSECT or a label definition (source ENTRY) defined in the first phase. This address is used as the transfer address to the first phase in the program. If the operand field is blank, the linkage editor uses as a transfer address the first significant address provided in an END record encountered during the generation of the first phase. If no such operand is found on the END card, the transfer address is the load address of the first phase.

It is necessary to supply the ENTRY statement only if a specific entry point is desired. Job control writes an ENTRY statement with a blank operand on SYSLNK when EXEC LNKEDT is read to ensure that an ENTRY statement will be present to halt link editing.

ACTION Statement

This statement is used to indicate linkage editor options. When used, the statement must be the first linkage editor record(s) in the input stream. If multiple operands are required, they can be placed in separate ACTION statements or in one ACTION statement separated by commas. Its format is:

```
ACTION {REL  }[,CLEAR][,MAP  ] [,NOAUTO]
       {NOREL}        [,NOMAP]
       [,CANCEL][,BG]
                [,Fn]
```

At least one blank must precede ACTION.

CLEAR — Indicates that the unused portion of the core image library will be set to binary zero before the beginning of the linkage editor function. CLEAR is a time-consuming function. It should be used only if it is necessary to fill areas defined by DS statements with zeros.

MAP — Indicates that SYSLST is available for diagnostic messages. In addition, a map of virtual storage is printed on SYSLST. The map contains the name of every entry within each CSECT and the name of every CSECT within each phase.

NOMAP — Indicates that SYSLST is not available when performing the link-edit function. Storage mapping is not performed and all linkage editor error diagnostics are listed on SYSLOG.

NOAUTO — Indicates the AUTOLINK function is to be suppressed during the link editing of the entire program. AUTOLINK will be suppressed for both the private and the system relocatable libraries.

Note: When a WX is encountered, it is treated in the same manner as an EXTRN, NOAUTO.

CANCEL Cancels the job automatically if any of the errors 2100I through 2170I occur. If this option is not specified, the job continues.

BG
F1
F2
F3
F4

Sets the end-of-supervisor address used in linkage editor calculations to the beginning of the virtual partition specified, plus the length of the label area and of the save area. The end-of-supervisor address in the communica- tion region is not changed.

The BG, F1, F2, F3, and F4 operands link edit a program to execute in a partition other than that in which the link edit function is taking place. Programs that have a phase origin of S (or * for the first phase of a program) are originated to the specified virtual partition.

Use of the ACTION BG statement is possible only in a system supporting more than one partition and the private core image library option when the linkage editor is executing in a foreground partition.

Use of the ACTION Fn statement in a multiprogramming environment requires that the partition be allocated. If these operands are used in a non-multiprogramming environment, they are ignored. If none of these operands are present, the program is link edited to execute in the virtual partition or in its associated real partition in which link-editing takes place, unless otherwise specified in the PHASE statement.

REL Indicates that the phase(s) produced during this execution of the linkage editor is (are) to be made relocatable, if possible. The format of the origin specified in the PHASE statement determines whether or not a relocatable phase can be produced. (Refer to the origin operand in the PHASE statement.) If support for the relocating loader was generated in the supervisor, ACTION REL is the default. At program execution time, a relocatable program can be loaded into any partition. For a more detailed description of the relocating loader and its use, refer to the DOS/VS System Management Guide, GC33-5371.

NOREL Indicates that the phase(s) produced during this execution of the linkage editor is (are) not to be made relocatable. If support for the relocating loader was not generated in the supervisor, ACTION NOREL is the default.

An ACTION statement flagged as invalid (as the result of an invalid operand, etc.), ignores all subsequent ACTION statements submitted during the job.

The ACTION statement is not required. If the MAP option is specified, SYSLST must be assigned. If the statement is not used and SYSLST is assigned, MAP is assumed and a map of virtual storage and any error diagnostics are printed on SYSLST. If the statement is not used and SYSLST is not assigned, NOMAP is assumed.

The following information is contained in the storage map.

1. The name of each phase, the lowest and highest virtual storage locations of each phase, and the hexadecimal disk address where the phase begins in the core image library.

2. An indication if the phase is a ROOT phase, or if a phase overlays the ROOT phase in any way (designated by OVEROOT).

3. The length of COMMON, if appropriate.

4. The names of all CSECTs belonging to a phase, the address where each CSECT is loaded, and the relocation factor of each CSECT.

5. All defined entry points within a CSECT. If an entry point is unreferenced, it is flagged with an asterisk (*).

6. The names of all external references that are unresolved.

7. The transfer (execute) address of each phase.

8. An indication that the phase is relocatable, self-relocating, not relocatable, or SVA eligible.

9. Warning messages are printed if:

   - The ROOT phase has been overlaid;

   - A possible invalid entry point duplication occurred;

   - The ENTRY or END statement contained an invalid (undefined) transfer label;

- At least one control section had a length of zero;

- The assembled origin on an RLD statement was outside the limits of the phase;

- An address constant could not be resolved.

These messages may or may not indicate actual programming errors. If NOMAP is operational, the warning messages are not printed.

The difference between specifying NOAUTO in a PHASE statement and specifying ACTION NOAUTO. The NOAUTO operand in a PHASE statement indicates to the Linkage Editor that AUTOLINK is to be suppressed for that phase only. If an entire program requires NOAUTO, then specifying ACTION NOAUTO cancels AUTOLINK during link editing of the entire program, thereby eliminating the necessity of specifying NOAUTO in each PHASE statement.

Figure 12 shows a storage map and a diagnostic listing produced on SYSLST. For the line numbers referred to in the following discussion, see Part 1 of Figure 12.

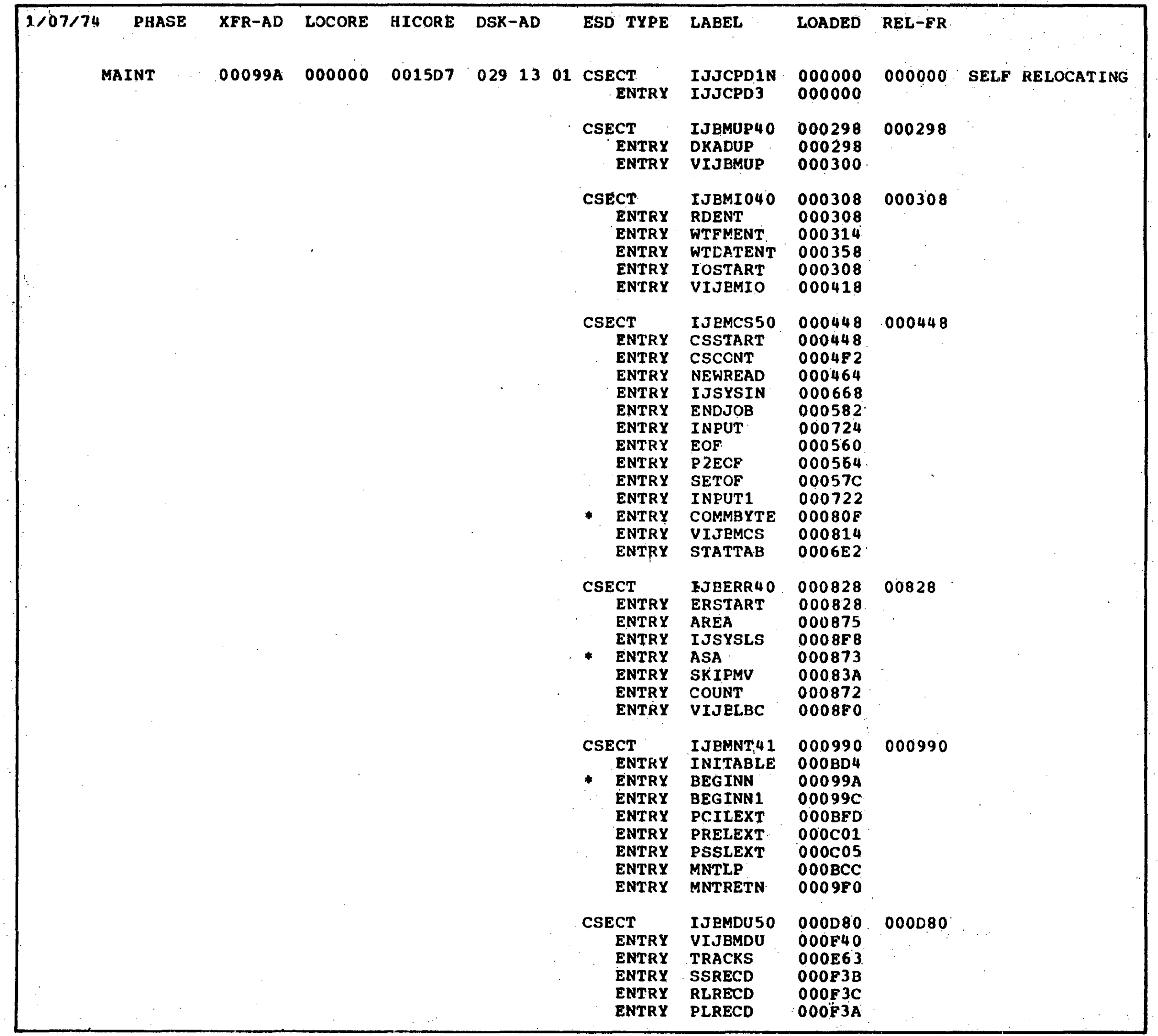
1. Line 1 (ACTION TAKEN). MAP and REL have been specified on separate ACTION cards. Had NOAUTO been specified, it would also appear on this line.

2. Lines 3, 13, 15, 17, 19, 21, 23, 25, and 27. These lines are printed when the statement is read by the linkage editor.

When a module is included from the relocatable library, it is not possible to guarantee that the sequence identification printed in columns 8-15 is that of the record printed. This occurs because the MAINT librarian program reblocks the content of the cards to a more compressed format.

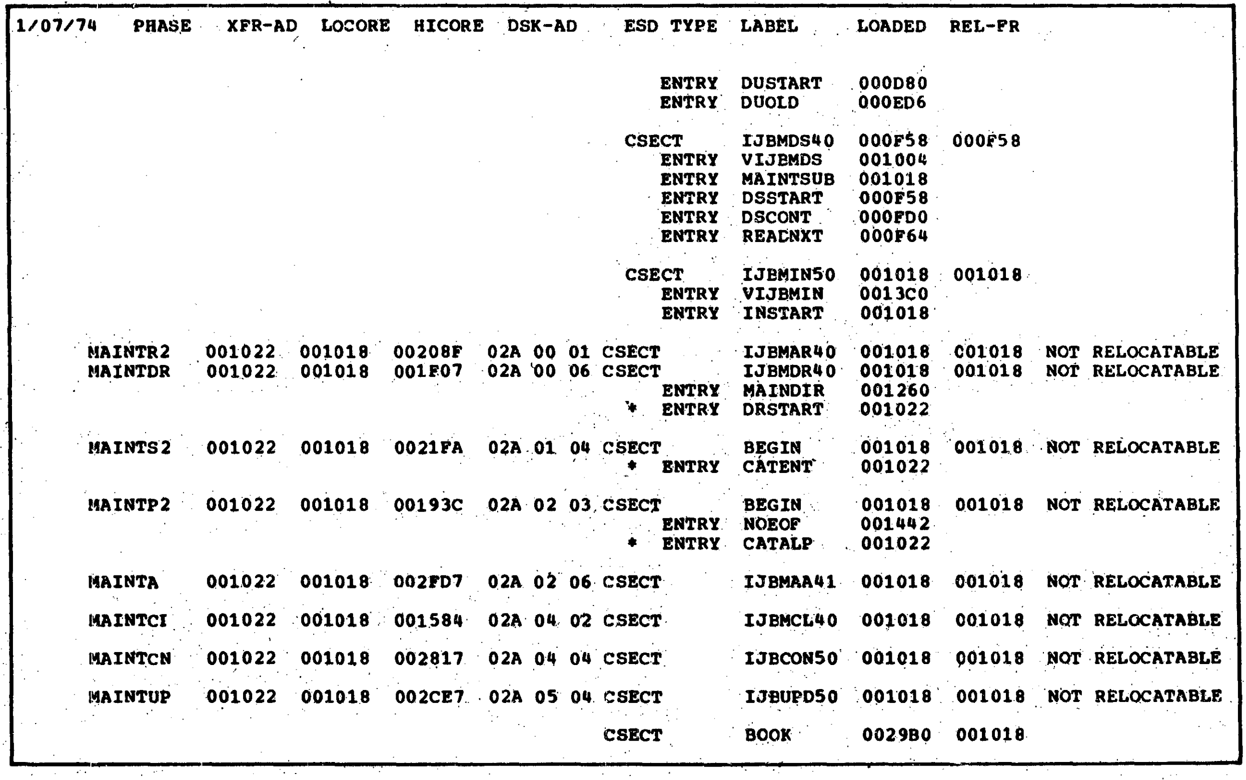
```
JOB  NO NAME   31/07/74    DOS LINKAGE EDITOR DIAGNOSTIC OF INPUT

ACTION TAKEN   MAP REL
LIST   INCLUDE IJBSL2                                                LNK00020
LIST           INCLUDE   IJBMIN              INITIALIZATION          03000000
LIST   PHASE MAINT,+0,NOAUTO
LIST   INCLUDE IJJCPD1N    LIOCS  I/O MODULE
LIST   INCLUDE IJBMUP      DISK ADDRESS UPDATE ROUTINE
LIST   INCLUDE IJBMIO      DISK I/O ROUTINE
LIST   INCLUDE IJBMCS      CARD SCAN ROUTINE
LIST   INCLUDE IJBLBC      LIBRARIAN ERROR ROUTINE
LIST   INCLUDE IJBLBA      MAINT FETCH ROUTINE
LIST   INCLUDE IJBMDU      DIRECTORY UPDATE ROUTINE
LIST   INCLUDE IJBMDS      DIRECTORY SCAN ROUTINE
LIST           INCLUDE    IJBLBE             MAINTR2                 06000000
LIST   PHASE MAINTR2,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBD             MAINTDR                 09000000
LIST   PHASE MAINTDR,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBF             MAINTS2                 12000000
LIST   PHASE MAINTS2,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBN             MAINTP2              PM 15000000
LIST   PHASE MAINTP2,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBL             MAINTA                  18000000
LIST   PHASE MAINTA,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBM             MAINTCL                 21000000
LIST   PHASE MAINTCL,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBG             MAINTCN                 27000000
LIST   PHASE MAINTCN,MAINTSUB,NOAUTO
LIST           INCLUDE    IJBLBQ             MAINTUP                 33000000
LIST   PHASE MAINTUP,MAINTSUB,NOAUTO
LIST   ENTRY
```

Figure 12. Storage Map (Part 1 of 3)



```
1/07/74   PHASE    XFR-AD  LOCORE  HICORE  DSK-AD     ESD TYPE   LABEL     LOADED  REL-FR


       MAINT       00099A  000000  0015D7  029 13 01  CSECT      IJJCPD1N  000000  000000  SELF RELOCATING
                                                         ENTRY   IJJCPD3   000000

                                                      CSECT      IJBMUP40  000298  000298
                                                         ENTRY   DKADUP    000298
                                                         ENTRY   VIJBMUP   000300

                                                      CSECT      IJBMIO40  000308  000308
                                                         ENTRY   RDENT     000308
                                                         ENTRY   WTFMENT   000314
                                                         ENTRY   WTDATENT  000358
                                                         ENTRY   IOSTART   000308
                                                         ENTRY   VIJBMIO   000418

                                                      CSECT      IJBMCS50  000448  000448
                                                         ENTRY   CSSTART   000448
                                                         ENTRY   CSCONT    0004F2
                                                         ENTRY   NEWREAD   000464
                                                         ENTRY   IJSYSIN   000668
                                                         ENTRY   ENDJOB    000582
                                                         ENTRY   INPUT     000724
                                                         ENTRY   EOF       000560
                                                         ENTRY   P2EOF     000564
                                                         ENTRY   SETOF     00057C
                                                         ENTRY   INPUT1    000722
                                                      *  ENTRY   COMMBYTE  00080F
                                                         ENTRY   VIJBMCS   000814
                                                         ENTRY   STATTAB   0006E2

                                                      CSECT      IJBERR40  000828  00828
                                                         ENTRY   ERSTART   000828
                                                         ENTRY   AREA      000875
                                                         ENTRY   IJSYSLS   0008F8
                                                      *  ENTRY   ASA       000873
                                                         ENTRY   SKIPMV    00083A
                                                         ENTRY   COUNT     000872
                                                         ENTRY   VIJBLBC   0008F0

                                                      CSECT      IJBMNT41  000990  000990
                                                         ENTRY   INITABLE  000BD4
                                                      *  ENTRY   BEGINN    00099A
                                                         ENTRY   BEGINN1   00099C
                                                         ENTRY   PCILEXT   000BFD
                                                         ENTRY   PRELEXT   000C01
                                                         ENTRY   PSSLEXT   000C05
                                                         ENTRY   MNTLP     000BCC
                                                         ENTRY   MNTRETN   0009F0

                                                      CSECT      IJBMDU50  000D80  000D80
                                                         ENTRY   VIJBMDU   000F40
                                                         ENTRY   TRACKS    000E63
                                                         ENTRY   SSRECD    000F3B
                                                         ENTRY   RLRECD    000F3C
                                                         ENTRY   PLRECD    000F3A
```

Figure 12. Storage Map (Part 2 of 3)



```
1/07/74 PHASE    XFR-AD  LOCORE  HICORE  DSK-AD    ESD TYPE  LABEL     LOADED  REL-FR

                                                       ENTRY DUSTART   000D80
                                                       ENTRY DUOLD     000ED6

                                                   CSECT     IJBMDS40  000F58  000F58
                                                       ENTRY VIJBMDS   001004
                                                       ENTRY MAINTSUB  001018
                                                       ENTRY DSSTART   000F58
                                                       ENTRY DSCONT    000FD0
                                                       ENTRY READNXT   000F64

                                                   CSECT     IJBMIN50  001018  001018
                                                       ENTRY VIJBMIN   0013C0
                                                       ENTRY INSTART   001018

        MAINTR2  001022  001018  00208F  02A 00 01 CSECT     IJBMAR40  001018  C01018  NOT RELOCATABLE
        MAINTDR  001022  001018  001F07  02A 00 06 CSECT     IJBMDR40  001018  001018  NOT RELOCATABLE
                                                       ENTRY MAINDIR   001260
                                                     * ENTRY DRSTART   001022

        MAINTS2  001022  001018  0021FA  02A 01 04 CSECT     BEGIN     001018  001018  NOT RELOCATABLE
                                                     * ENTRY CATENT    001022

        MAINTP2  001022  001018  00193C  02A 02 03 CSECT     BEGIN     001018  001018  NOT RELOCATABLE
                                                       ENTRY NOEOF     001442
                                                     * ENTRY CATALP    001022

        MAINTA   001022  001018  002FD7  02A 02 06 CSECT     IJBMAA41  001018  001018  NOT RELOCATABLE

        MAINTCI  001022  001018  001584  02A 04 02 CSECT     IJBMCL40  001018  001018  NOT RELOCATABLE

        MAINTCN  001022  001018  002817  02A 04 04 CSECT     IJBCON50  001018  001018  NOT RELOCATABLE

        MAINTUP  001022  001018  002CE7  02A 05 04 CSECT     IJBUPD50  001018  001018  NOT RELOCATABLE

                                                   CSECT     BOOK      0029B0  001018
```

Figure 12. Storage Map (Part 3 of 3)

Linkage Editor Input Considerations

The storage requirements for a link-edit run depend on:

- Input to the linkage editor (number of PHASE statements and number of ESD items).
- Whether $MAINDIR (the phase that is called by the linkage editor and librarian programs to maintain the core image directory) runs in the SVA or not.

  Note: It is assumed that $LIBSTAT (the phase that prints out the status report of the libraries) runs in the SVA if $MAINDIR does.
- The start address of the linkage editor during execution. (The work area for the linkage editor starts on a 1K boundary, while the linkage table starts on a page boundary. In the following storage estimate formula, the worst case is assumed, that is, the maximum storage amount of 2K is used to ensure correct alignment.)

In a virtual partition of 64K with $MAINDIR running in the SVA the linkage editor can process at least 50 phases with a total number of 1700 unique ESD items.

A unique ESD item is defined as being an occurrence in the control dictionary. All symbols that appear in the MAP are unique occurrences. A symbol that occurs several times in the input stream is normally incorporated into a unique ESD item. However, if the same symbol occurs in different phases (for example, control sections), each resolved occurrence of the symbol within a different phase is a unique ESD item.

You can use the following formula for storage estimates. If the relation is satisfied, then you have enough address space for the link-edit run.

$$26{,}900 + M + ST + 52 * x + 20 * y \le P$$

where M = 0 if $MAINDIR runs in SVA
10,240 if $MAINDIR runs in partition

ST = 30 * x if 30 * x < 2048
2048 if 30 * x ≥ 2048

x = number of PHASE statements

y = total number of unique ESD items

P = available storage in partition.

Linkage Editor Job Setup

When link editing in the foreground, a private core image library (SYSCLB) must be uniquely assigned to the partition. The program phase (output of the linkage editor) is put in the private core image library. In the background, the linkage editor can put a phase in either a private or the system core image library. If SYSCLB is uniquely assigned in the background, the linkage editor output is put in the private core image library. If SYSCLB is not assigned, the system core image library is used.

When performing a link-edit function, the following system and programmer logical units are used. SYSRDR and SYSIPT may contain input for the linkage editor. This input is written onto SYSLNK by job control.

Unit	Function
SYSRDR	Control statement input (via Job Control)
SYSIPT	Module input
SYSLST	Programmer messages and listings
SYSLOG	Operator messages
SYSLNK	Input to the Linkage Editor
SYS001	Workfile

In normal operations, all preceding logical units must be assigned. In a unique circumstance (when all modules to be link-edited are in the relocatable library), SYSIPT would not need to be assigned.

If output from the linkage editor is to be placed in a private core image library, the following symbolic unit is also required:

SYSCLB The private core image library

A link-edit job is set up in the following manner.

Control Statement	Remarks
// JOB	Required only if this is the first job step of a job.
// ASSGN	Required only if device assignments are to differ from the system standard assignments. Units that can be assigned are SYSRDR, SYSIPT, SYSLST, SYSLNK, and SYS001.

ASSGN SYSCLB — Required if output of linkage editor is to be placed in a private core image library.

// OPTION — OPTION statement must follow the ASSGN statement (if any) for SYSLNK.

ACTION — Optional ACTION statement (with appropriate operands) must precede the first linkage editor control statement.

PHASE INCLUDE — As many PHASE and INCLUDE statements as are required are used to construct phases from the modules input to the linkage editor.

ENTRY — Optional statement to provide a transfer address for the first phase.

// LBLTYP — LBLTYP statement (if required) to define the amount of storage to be reserved at link-edit time for processing of tape or nonsequential DASD file labels in the virtual storage partition.

// EXEC LNKEDT — EXEC statement to call the linkage editor from the core image library. Job control creates an ENTRY statement on SYSLNK to ensure its presence to halt link editing.

/& — End-of-job statement.

When link-editing multiple object modules into one program phase, make sure that the linkage editor selects the intended entry point. Either specify or place the main control section first in the linkage editor input, or use a linkage editor ENTRY statement with the name of the main control section as the <u>entry-point</u> operand.

Example of Linkage Editor Input and Output

The program shown in Figure 13 illustrates the rules governing input for the linkage editor and shows the output obtained. Though this example is somewhat more complex than the normal program, by following the flow of the input, one can find practically every situation that may arise.

The leftmost block shows control statements being read by job control from SYSRDR. The next block is read by job control from SYSIPT and contains an object module (module 1) and a source module to be assembled. The next block shows the output from job control on SYSLNK, which is the input to the linkage editor. The next two blocks represent two levels in the relocatable library. The rightmost block shows the output phases as they appear in the temporary portion of the core image library after the execution of the linkage editor function. A detailed sequence of events follows.

Linkage editor control statements are read by job control from SYSRDR and are copied on SYSLNK until an INCLUDE statement with a blank first operand is read. This statement is not copied on SYSLNK. Instead, job control copies the module on SYSIPT onto SYSLNK until a /* statement is read. Job control then reads from SYSRDR. An assembly is executed and its output is written directly on SYSLNK. (It is assumed that LINK was specified in an OPTION statement preceding the linkage editor control statements.) Job control then writes the ENTRY statement with a transfer label for CS A on SYSLNK and issues a fetch for the linkage editor.

The linkage editor reads from SYSLNK and starts to create a program. An INCLUDE statement with a first operand signals the linkage editor to access the relocatable library. This is the first level of an INCLUDE. In the first level of the relocatable library, the linkage editor reads an INCLUDE (for the second level) and performs this inclusion. As no INCLUDE is present in the second level, control is returned to the calling input level. This process is repeated for the next INCLUDE. Note that the namelist specifies only CS C is wanted.

After the inclusion of the module at the first level, control is returned to SYSLNK where a new phase is encountered. The control sections are read from SYSLNK and added to PHASE2 until the next INCLUDE is read. At this time, the linkage editor again accesses the relocatable library, performs the inclusion of MOD5 into PHASE2, and continues reading input from SYSLNK. Processing continues until the ENTRY statement is reached.

The split control section (CS E) is assigned a contiguous area of virtual storage.

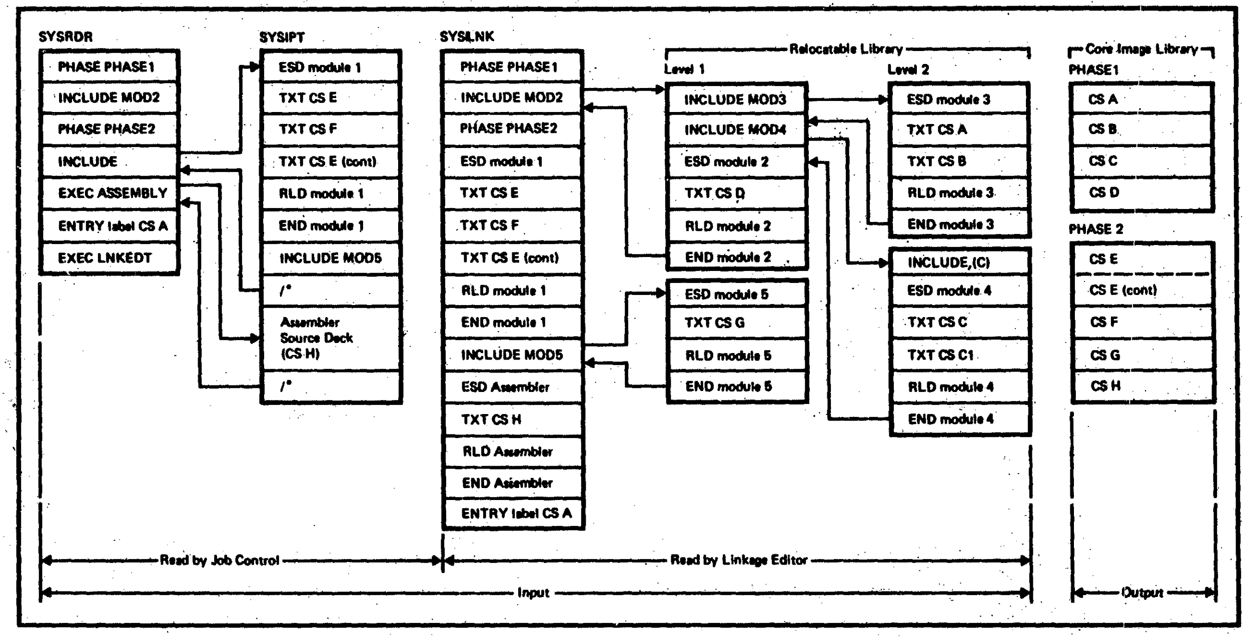


Figure 13. Example of Linkage Editor Input and Output

Librarian

This section describes the set of programs that maintain, service, and copy the libraries of DOS/VS. This set of programs is collectively referred to as the librarian.

The system residence (SYSRES) can contain four separate and distinct system libraries:

1. Core image library
2. Relocatable library
3. Source statement library
4. Procedure library.

The core image library is the only library that is required.

Figure 14 lists the logical units required and used by the librarian programs.

	MAINT	CORGZ	DSERV	CSERV	RSERV	SSERV	PSERV	ESERV
SYSRDR[1]	R	R	R	R	R	R	R	
SYSIPT	R	R	R	R	R	R	R	R
SYSLST	O	R	R	O	O	O	O	O
SYSLOG[2]	R	R	R	R	R	R	R	R
SYSRES	R	R	R	R	R	R	R	R
SYSCLB	O	O	O	O				
SYSRLB	O	O	O		O			
SYSSLB	O	O	O			O		O
SYS000		O						
SYS001		O						
SYS002		O						
SYS003		O						
SYSIN[3]	O	O	O	O	O	O	O	O
SYSPCH	O			O	O	O	O	O

R = Required
O = Optional dependent upon function specified

[1] SYSRDR is required for job control purposes only.
[2] SYSLOG must not be assigned to a printer
[3] If SYSRDR and SYSIPT are assigned to the same device, SYSIN may be used.

Figure 14. Logical Units Required and Used by the Librarian Programs

		Directory			
		Core Image	Relocatable	Source Statement	Procedure
NUMBER OF ENTRIES:					
2311*			--	--	--
2314/2319	first track	Variable (see Note)	335 module	265 book	265 procedure
	other tracks		340 entries	270 entries	270 entries
	last track		339	269	269
3330/3333	first track		555 module	435 book	435 procedure
	other tracks		560 entries	440 entries	440 entries
	last track		559	439	439
3340	first track		335 module	255 book	255 procedure
	other tracks		340 entries	260 entries	260 entries
	last track		339	259	259
NUMBER OF TRACKS:					
2311*			--	--	--
2314/2319		Variable (see Note)	TRD = $\frac{m+6}{340}$	TSD = $\frac{b+6}{270}$	TPD = $\frac{d+6}{270}$
3330/3333			TRD = $\frac{m+6}{560}$	TSD = $\frac{b+6}{440}$	TPD = $\frac{d+6}{440}$
3340			TRD = $\frac{m+6}{340}$	TSD = $\frac{b+6}{260}$	TPD = $\frac{d+6}{260}$

m = Total number of modules in the relocatable library.
b = Total number of books in the source statement library.
d = Total number of procedures in the procedure library.

The value TxD is rounded to the next higher integer if a remainder results.

*The 2311 is supported as a private core image library only.

Note: The entries in the core image directory are of variable length with a maximum length of 30 bytes. The length of the entries is determined by the information contained in them. The load point at link-edit time, the entry point at link-edit time, the number of RLD items, the number of additional RLD blocks, and the partition start address at link-edit time are only part of the directory entry under certain conditions. The number of blocks (256 bytes) per track is 10 for the 2311 (only private core image library), 17 for the 2314/2319, 28 for the 3330/3333, and 16 for the 3340.

Figure 15. Calculation of the Tracks Required for Directories

Disk Storage Space Required for Libraries and Directories

You must determine the amount of space allocated to each of the libraries and directories. Each library consists of one or more complete disk tracks. Each directory occupies the first track(s) of the first cylinder allocated to its respective library.

DIRECTORY SIZES

Figure 15 summarizes the track requirements for the core image, relocatable, source statement, and procedure directories.

CORE IMAGE LIBRARY SIZE

Each track allocated to the core image library contains three fixed-length blocks on 2311, six fixed-length blocks on

2314/2319, eleven fixed-length blocks on 3330/3333, or seven fixed-length blocks on 3340. Each block contains 1024 bytes of instructions or data. For non-relocatable phases, the core image library contains exactly the same information as is loaded into virtual storage for execution. For relocatable phases, the CIL contains both TXT and RLD information. Each phase is written beginning in a new block. The number of tracks required for the core image library can be calculated as follows:

1. Determine the number of blocks (B) required for a phase:

   $$B = \frac{L}{1024}$$

   where L is the total number of bytes in the phase (rounded up to a multiple of 4) plus 4 times the number of RLD items (see the assembly listing). The value B is rounded to the next higher integer if a remainder results.

2. Determine the total number of blocks (Bt) required for all phases in the core image library:

   $$Bt = B1 + B2 + B3 + ... + Bn$$

3. Determine the number of tracks (TCL) required to hold all phases in the core image library:

   $$TCL = \frac{Bt}{n}$$

   where n = 3 for the 2311 (supported as a private core image library only)
   6 for the 2314/2319
   11 for the 3330/3333
   7 for the 3340

4. Determine the number of cylinders (CCL) required to hold the core image library and core image directory:

   $$CCL = \frac{TCD+TCL}{n}$$

   where n = 10 for the 2311 (supported as a private core image library only)
   20 for the 2314/2319
   19 for the 3330/3333
   12 for the 3340

   The value of CCL is rounded to the next higher integer if a remainder results. This calculation is useful when executing MAINT and CORGZ. When allocating a SYSRES file, add two tracks for system purposes.

RELOCATABLE LIBRARY SIZE

Each track allocated to the relocatable library contains 16 fixed-length blocks for the 2314/2319, 28 fixed-length blocks for the 3330/3333, or 17 fixed-length blocks for the 3340. Each block is 322 bytes long. A number of factors affects the packing of information in these blocks. The factors include the following variables:

1. The number of separate control sections.

2. The use of DS (define storage) statements, which reserve storage that may or may not be utilized for data constants defined in the program.

3. Alternation of the location counter during assembly (use of ORG statements).

The following calculations approximate fairly accurately the library area required for typical programs.

1. Determine the number of blocks (Bc) required for all cards or statements except the actual program text. Assume a separate block for each card of the following types:

   a. PHASE
   b. INCLUDE
   c. REP
   d. END
   e. SYM
   f. ENTRY

   Let Bc equal the total number of cards of the above types.

2. Determine the number of blocks (Be) required for ESD and RLD cards. Assume a separate block for every two ESD or RLD cards.

3. Determine the number of blocks (Bi) required for the actual instructions or data in the TXT cards. Assume an average of 200 bytes of text in each block. (A maximum per block, for continuously assigned text, is 264 bytes per block.) Thus:

   $$Bi = \frac{\text{total bytes of text in TXT cards}}{200}$$

4. Determine the total number of blocks (Bn) required for a module in the relocatable library:

   $$Bn = Bc + Bi + Be$$

5. Determine the total number of blocks (Bt) required to hold all of the modules in the library:

   $$Bt = B1 + B2 + B3 + ... + Bn$$

6. Determine the number of tracks (TRL)

   $TRL = \frac{Bt}{n}$

   where n = 16 for the 2314/2319
   28 for the 3330/3333
   17 for the 3340

   The value TRL is rounded to the next higher integer if a remainder results.

7. Determine the number of cylinders (CRL) required to hold the relocatable library and directory:

   $CRL = \frac{TRD+TRL}{n}$

   where n = 20 for the 2314/2319
   19 for the 3330/3333
   12 for the 3340

   The value CRL is rounded to the next higher integer if a remainder results.

SOURCE STATEMENT LIBRARY SIZE

Each track allocated to the source statement library contains 27 fixed-length blocks for the 2314/2319 or 44 fixed-length blocks for the 3330/3333 or 26 fixed-length blocks for the 3340). Each block contains a maximum of 160 bytes of source statement information. The source statements are compressed before they are written out on the source statement library. This compression is performed by eliminating all blanks in each source statement. Several count bytes indicating the number of blanks eliminated are added to each statement before writing it in the source statement library. The number of tracks required for the source statement library can be calculated as follows:

1. Determine the number of statements (N) used to define a book. If the book is an edited macro definition produced by the assembler, determine the number of statements in the <u>source format</u> definition.

2. Determine the average compressed statement length (Ls) in the book. The compressed statement length approximately equals:

   $Ls = (L1+1) + \ldots. + (Ln+1) + 3$

   where each Ln is the number of bytes in each word of the source statement. For the macro sublibrary (sublibrary E) multiply the result by 1.2.

3. Determine the number of blocks (Bn) needed to hold the book:

   $Bn = \frac{N(Ls)}{160}$

   The value Bn is rounded to the next higher integer if a remainder results.

4. Determine the total number of blocks (Bt) required to hold all books in the library:

   $Bt = B1 + B2 + B3 + \ldots. + Bn$

5. Determine the number of tracks (TSL) required to hold all of the books in the source statement library:

   $TSL = \frac{Bt}{n}$

   where n = 27 for the 2314/2319
   44 for the 3330/3333
   26 for the 3340

   The value TSL is rounded to the next higher integer if a remainder results.

6. Determine the number of cylinders (CSL) required to hold the source statement library and source statement directory:

   $CSL = \frac{TSL+TSD}{n}$

   where n = 20 for the 2314/2319
   19 for the 3330/3333
   12 for the 3340

   The value CSL is rounded to the next higher integer if a remainder results.

PROCEDURE LIBRARY SIZE

Each track allocated to the procedure library contains 40 fixed-length blocks for the 2314/2319, 61 fixed-length blocks for the 3330/3333, or 34 fixed-length blocks for the 3340. Each block is 80 bytes long. A cataloged procedure consists of one or more consecutive blocks, each containing a control statement or data, in uncompressed card-image format. The first five blocks of the procedure library are save areas for currently uninterpreted overwrite statements (1 block/partition) that have already been read. The number of tracks required for the procedure library can be calculated as follows:

1. Determine the number of blocks (B) for a procedure:

   B = N

   where N is the number of control statements and data cards in the procedure.

2. Determine the total number of blocks (Bt) required to hold all the procedures in the library, and add 5 for the save areas:

   Bt = B1 + B2 + B3 +..., Bn + 5

3. Determine the number of tracks (TPL) required to hold all the procedure library:

   $TPL = \frac{Bt}{n}$

   where n = 40 for the 2314/2319
   61 for the 3330/3333
   34 for the 3340

   The value of TPL is rounded to the next higher integer if a remainder results.

4. Determine the number of cylinders (CPL) required to hold the procedure library and procedure directory:

   $CPL = \frac{TPD+TPL}{n}$

   where n = 20 for the 2314/2319
   19 for the 3330/3333
   12 for the 3340

   The value of CPL is rounded to the next higher integer if a remainder results.

GENERAL CONTROL STATEMENT FORMAT

The librarian control statements are similar in format to statements processed by the assembler. The operation field must be preceded by one or more blanks. The operation field must begin to the right of column 1 and must be separated from the operand field by at least one blank position. The operand field is terminated by the first blank position. It cannot extend past column 71. Continuation statements are not recognized.

Maintenance Program

This section describes the control statements that can be submitted to the MAINT program, which maintains the core image, relocatable, source statement, and procedure libraries. Except for the reallocation function of MAINT, all control statements apply to both system and private libraries; the reallocation function, however, cannot be used for private libraries.

To invoke the maintenance program, use the following job control statement:

```
// EXEC MAINT
```

One or more of the maintenance functions for one or more of the libraries can be requested within a single run:

- catalog
- delete
- rename
- condense
- reallocate
- update

Any number of phases, modules, books, or procedures can be acted upon in one run.

Note: If POWER/VS or any unending job is started with a cataloged procedure, no maintenance functions can be performed on the procedure library as long as the job is active.

The following symbolic units must be assigned to perform maintenance functions on a private library: SYSCLB for a private core image library, SYSRLB for a private relocatable library, and SYSSLB for a private source statement library. A private library must be unassigned to perform maintenance functions on the corresponding system libraries.

CATALOG

The catalog function, which adds elements to a library, is described in the following sections:

CATALR -- to catalog to the relocatable library
CATALS -- to catalog to the source statement library
CATALP -- to catalog to the procedure library.

Note: To catalog to the core image library, refer to the chapter Linkage Editor.

CATALR

The CATALR control statement is used to add a module to the relocatable library. The CATALR control statement is read from the device assigned to SYSIPT and is in the following format:

```
CATALR modulename[,v.m]
```

The operation field contains CATALR. The entry in the operand field, modulename, is the name by which the module is to be known to the control system. The modulename is one to eight characters, the first of which must not be an asterisk.

The optional entry in the operand field, v.m, specifies the change level at which the module is to be cataloged. v may be any decimal number from 0-127. m may be any decimal number from 0-255. If this operand is omitted, a change level of 0.0 is assumed.

A change level can be assigned only when a module is cataloged. The change level is displayed and punched by the service functions.

The statements composing the input for a module are described in the Linkage Editor section. The statements are:

1. PHASE
2. INCLUDE control statement (if appropriate)
3. ESD
4. TXT
5. RLD
6. REP
7. END
8. ENTRY

These statements are read from the device assigned to SYSIPT. All input is diagnosed by the linkage editor. The CATALR statement is recognized but ignored by the linkage editor. The END statement indicates end of module.

The ENTRY statement can only be used in a module that contains only linkage editor control statements and an END statement. The ENTRY statement must be the last control statement in the module, following the END statement.

Normally, modules in the relocatable library are output from a language translator. However, you can construct an artificial module of linkage editor control statements, referred to as a calling module. The following example illustrates a valid calling module:

```
PHASE PHNAM1,ROOT
INCLUDE MODULE1
PHASE PHNAM2,*
INCLUDE MODULE2
PHASE PHNAM3,PHNAM2
       .
       .
       .
ENTRY CSECTNME
END
```

Operands in INCLUDE statements refer to modules in the relocatable library. If, for example, the preceding calling module is cataloged by the name BIGPROG, all modules referred to in BIGPROG can be link-edited by using the following control statements:

```
// OPTION CATAL
   INCLUDE BIGPROG
// EXEC LNKEDT
```

A calling module may consist only of INCLUDE statements. In this case, the PHASE statements would precede the included modules.

A ninth statement, SYM, can be in the linkage editor input. When recognized, however, it is bypassed by the linkage editor. (The SYM statement identifies the symbol table output by the assembler as a result of specifying SYM in the OPTION statement).

For the catalog function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer. If SYSIPT is assigned to a tape unit, the MAINT program assumes that the tape is positioned to the first input record. The tape is not rewound at the end of job.

Any number of modules can be cataloged in a single run. Each module must immediately follow its respective CATALR control statement.

An additional capability of the system allows for assembling or compiling a program and cataloging it to a relocatable library in one continuous run. This is done by inserting a CATALR statement in the job control input stream preceding the phase statement (if present) and the assembler/compiler execute statement. The CATALR statement is written on the SYSPCH file, if SYSPCH is assigned, (on tape, DASD, or 3540 diskette) ahead of the assembler/compiler output. Then reassign the SYSPCH file as SYSIPT and execute the MAINT program to perform the catalog function. The output of the assembly/compilation (on tape or DASD, or 3540 diskette) may be cataloged immediately or it may be cataloged at some later time. It can also be held after cataloging as backup of the assembly/compilation.

CATALS

The CATALS control statement is used to add a book to a sublibrary of the source statement library. It is read from the device assigned to SYSIPT and is in the following format.

```
CATALS sublib.bookname[,v.m[,C]]
```

The operation field contains CATALS. The qualifier sublib in the operand field represents the sublibrary to which the book is to be cataloged and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The sublib qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the book.

bookname in the operand field represents the name of the book to be cataloged. The bookname is one to eight alphameric characters (0-9, A-Z, #, $, and @), the first of which must be alphabetic (A-Z, #, $, and @).

The first optional entry in the operand field, v.m, specifies the change level at which the book is to be cataloged. v may be any decimal number from 0-127. m may be any decimal number from 0-255. If this operand is omitted, a change level of 0.0 is assumed. The change level is displayed and punched by the service functions.

The second optional entry in the operand field, C, indicates that change level verification is required before updates are accepted for this book, providing the v.m operand is present on the update card (see UPDATE librarian control statement). This requirement is reflected in the DSERV output by a C appearing in the column headed LEV CHK (level check).

Books that are to be cataloged in a sublibrary of the source statement library must be preceded and followed by special statements indicating the beginning and the end of a book.

Macro definitions that are to be cataloged in the assembler sublibrary are preceded by the MACRO statement and are followed by the MEND statement. MACRO is the standard macro definition header statement; MEND is the standard macro definition trailer statement.

When books to be retrieved by the assembler COPY statement are to be cataloged to the assembler sublibrary, the assembler END statement should not be included in the book. (Assembler does not recognize END statements from the source statement library.)

Books other than macro definitions that are to be cataloged in the source statement library are preceded and followed by a BKEND statement. A BKEND statement must precede each book, and a BKEND statement must follow each book. If desired, the BKEND statement may precede and follow a macro definition (in addition to the MACRO and MEND statements). This is desirable when the options provided in the BKEND statement are required. The statement is in the following format.

```
BKEND [sub.book],[seq-chk],[count],[CMPRSD
```

The entry in the operation field is BKEND. All operand entries are optional. When used, the entries must be in the prescribed order, and need appear only in the BKEND statement preceding the book to be cataloged. The first entry in the operand field, sub.book, is identical to the operand of the CATALS control statement. The second operand seq-chk specifies that sequence checking is required. Which columns will be checked depends on how this operand is specified. Columns 73-78 will be checked if SEQNFS is specified. Columns 77-80 will be checked if SEQNCE is specified. The count operand specifies the number of card images in the book. When used, the card input is counted, beginning with the preceding BKEND statement and including the following BKEND statement. If an error is detected in either the sequence checking or the card count, an error message is printed. The error can be corrected, and the book can be recataloged. The CMPRSD operand indicates that the book to be cataloged in the library is in the compressed format, output as a result of specifying CMPRSD when performing a PUNCH or DSPCH service function.

For the catalog function SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, or a disk unit, and SYSLOG must not be assigned to a printer.

Any number of books may be cataloged in a single run. Each book must immediately follow its respective CATALS control. The function can also delete an entire sublibrary or an entire library.

CATALP

The CATALP control statement is used to add a procedure to the procedure library. Any number of procedures may be cataloged in a single run. Each procedure must immediately follow the respective CATALP statement.

Statement Format:

```
CATALP procedurename [,VM=v.m][,EOP=yy]
       [,DATA=NO ]
       [      YES]
```

Each control statement in the procedure library should have a unique identity. This identity is required if you want to modify the job stream at execution time. Therefore, when cataloging, identify each control statement in columns 73-79 (blanks may be embedded).

procedurename
: represents the name of the procedure to be cataloged. The procedurename consists of one to eight alphabetic and/or numeric characters. The name of the procedure must not be ALL and must not begin with $$ or with a numeric character.

Procedurenames can be related to the partition in which the procedure is intended to be run. A partition-related procedurename must follow these conventions:

$B for the BG partition
$1 for the F1 partition
$2 for the F2 partition
$3 for the F3 partition
$4 for the F4 partition.

VM=v,m specified the change level at which the procedure is to be cataloged. v may be any decimal number from 0-127. m may be any decimal number from 0-255. If this operand is omitted, a change level of 0.0 is assumed.

A change level can be assigned only when a procedure is cataloged. The change level is displayed and punched by the service functions.

EOP=yy specifies a two-character end-of-procedure delimiter. The EOP parameter can be any combination of characters except /*, /&, //; it must not contain a blank or a comma. The system assumes /+ as default end-of-procedure delimiter. Otherwise, you can omit the EOP parameter.

DATA=YES specifies that a procedure contains SYSIPT inline data. These procedures can only be executed in the extended procedure support.

A procedure to be cataloged into the procedure library may consist of job control and linkage editor statements and, if the supervisor was generated with the SYSFIL option, any additional SYSIPT data which must be processed under control of a device-independent sequential IOCS module (for instance, control statements for utility programs and service programs, source statements for compilers and object modules for the linkage editor). The end of a procedure is indicated by the /+ end-of-procedure delimiter or by the end-of-procedure delimiter as specified in the EOP parameter.

For the catalog function SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer. If SYSIPT is assigned to a tape unit, the MAINT program assumes that the tape is positioned to the first input record. The tape is not rewound at end of job.

A sample job stream follows.

```
// JOB CATPROC
   .
   .
   ASSGN control statement,
   if required
   .
   .
// EXEC MAINT
   CATALP PROCA,EOP=/$,DATA=YES
   .
   .
   control statements
   .
   .
   SYSIPT inline data
   .
   .
/* END OF SYSIPT DATA
   .
   .
   control statements
   .
   .
/$ END OF PROCEDURE
/&
```

The following restrictions apply when you catalog procedures to the procedure library:

1. A cataloged procedure cannot contain control statements or SYSIPT data for more than one job.

2. If the cataloged control statements include the JOB statement, you must not have a JOB statement when you retrieve the procedure through the EXEC statement. Conversely, if the JOB statement is not cataloged, a JOB statement must precede the EXEC statement that retrieves the procedure.

3. A cataloged procedure must not include any of the following control statements because they are not accepted when the procedure is processed:

```
// ASSGN SYSRDR,X'cuu'
// RESET SYS
// RESET ALL
// RESET SYSRDR
// CLOSE SYSRDR,X'cuu'
// ASSGN SYSIPT,X'cuu'
// RESET SYSIPT  only if SYSIPT data is
   included.
// CLOSE SYSIPT,X'cuu'
/&
```

4. Cataloged procedures cannot be nested, that is, a cataloged procedure cannot contain an EXEC statement that invokes another cataloged procedure.

DELETE

The delete function, which removes references to specific elements of a library, is described in the following sections:

DELETC -- to delete from the core image library

DELETR -- to delete from the relocatable library

DELETS -- to delete from the source statement library

DELETP -- to delete from the procedure library.

DELETC

The DELETC control statement in one of the following formats is used to delete phases or programs from the core image library.

```
DELETC phasename1[,phasename2,...]

DELETC prog1.ALL[,prog2.ALL,...]
```

In the first format, the entry in the operation field is DELETC. phasename in the operand field represents the name(s) of the phase(s) to be deleted. The name of the phase may be a maximum of eight characters. Entries in the operand field must be separated by commas.

In the second format, prog refers to the first four characters of the program name. (All phases within a program have the same first four characters. Therefore, the first four characters of each program within the library should be unique.) The four characters are followed by a period and ALL.

Any number of DELETC control statements can be used for the core image library within a single run.

For the delete function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

DELETR

The DELETR control statement in one of the following formats is used to delete a module from the relocatable library.

```
DELETR modname[,modname,...]

DELETR prog1.ALL[,prog2.ALL,...]

DELETR ALL
```

The first format is used when a specific module is to be deleted, and must always be used when deleting DIMODs. The entry in the operation field is DELETR. The entry in the operand field, modname, is the name of the module to be deleted. If more than one module is to be deleted, the module names are separated by a comma. modname is one to eight characters, the first of which must not be an asterisk.

The second format is used when an entire program is to be deleted. The entry in the operation field is DELETR. In the operand field, prog refers to the first three characters of the modules used to build the program. (All IBM-supplied modules in the relocatable library making up a program have the same first three characters, such as IPK for the assembler and IJS for COBOL.) The three characters are followed by a period and ALL.

The third format is used if the entire library is to be deleted. The entry in the operation field is DELETR. The entry in the operand field is ALL. When this function is performed, the system status record is reset to show that all library blocks are now available to the system. Therefore, it is unnecessary to perform a condense function after a DELETR ALL has been performed.

Any number of DELETR control statements can be used for the relocatable library within a single run.

For the delete function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

DELETS

The DELETS control statement is used to delete books from the source statement library. The control statement is in one of the following formats.

```
DELETS sublib.book1[,sublib.book2,...]

DELETS sublib.ALL

DELETS ALL
```

The first format is used if only specific books are to be deleted. The entry in the operation field is DELETS. The qualifier sublib in the operand field represents the sublibrary containing the book to be deleted and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The sublib qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the book.

book in the operand field represents the name of the book in the sublibrary to be deleted. If more than one book is to be deleted, the entries must be separated by commas. If books to be deleted are in the same sublibrary, subsequent book names need not be qualified. (The librarian assumes that nonqualified books are in the last sublibrary specified.) The name of the book can be of any length; however, a maximum of the first eight characters is used to locate and delete the book. Continuation statements are not recognized.

The second format is used if an entire sublibrary is to be deleted. The entry in the operation field is DELETS. The first entry in the operand field is the name of the sublibrary to be deleted. The qualifier sublib represents the sublibrary containing the book to be deleted and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The sublib qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the sublibrary.

The second entry in the operand field is ALL. The two entries must be separated by a period.

The third format is used if the entire source statement library is to be deleted. The entry in the operation field is DELETS. The entry in the operand field is ALL. When this function is performed, the system status record is reset to show that all library blocks are now available to the system. Therefore, it is unnecessary to perform a condense after a DELETS ALL has been performed.

For the delete function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

DELETP

The DELETP control statement in one of the following formats is used to delete a procedure from the procedure library:

```
{DELETP procedurename[,procedurename,...]}
{DELETP ALL                               }
```

procedurename
specifies the name of the procedure to be deleted. If more than one procedure is to be deleted, the names are separated by commas. The procedure name is one to eight alphameric characters, the first of which must be alphabetic or $.

ALL
specifies that the entire library is to be deleted. After a DELETP ALL has been performed, the system status record is reset to show that all library blocks are now available to the system. Therefore, it is unnecessary to perform a condense function after a DELETP ALL has been performed.

Any number of DELETP control statements can be used for the procedure library within a single run.

For the delete function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

RENAME

The rename function, which changes the name of an element in a library, is described in the following sections:

RENAMC -- to rename a phase in the core image library

RENAMR -- to rename a module in the relocatable library

RENAMS -- to rename a book in the source statement library

RENAMP -- to rename a procedure in the procedure library.

RENAMC

The RENAMC control statement is used to change the name of a phase in the core image library to another name. If the new name is already in the directory or an old name is not in the directory, an error message is issued. On a valid pair of operands, the new name simply replaces the old name in the directory; the version and modification levels are not changed. In either case, a check is then made for more operands on the card. As soon as the /& statement is processed, only the new phase names exist in the directory. The RENAMC statement is in the following format.

```
RENAMC oldname,newname[,oldname,newname,..]
```

The operation field contains RENAMC. The operand field entries, <u>oldname</u> and <u>newname</u>, represent the old phase name and the new phase name. The two entries in the operand field must be separated by a comma. The names in the operand field may be a maximum of eight characters. <u>Note</u>: ALL is an invalid phase name.

Any number of RENAMC control statements can be used for the core image library within a single run.

For the rename function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

RENAMR

The RENAMR control statement is used to change the name of a module in the relocatable library to another name. If the new name is already in the directory or an old name is not in the directory, an error message is issued. On a valid pair of operands, the new name simply replaces the old name in the directory; the version and modification levels are not changed. In either case, a check is then made for more operands on the card. As soon as the statement is processed, the system recognizes only the new module name. The RENAMR statement is in the following format:

```
RENAMR oldname,newname[,oldname,newname,..]
```

The operation field contains RENAMR. The entries in the operand field, <u>oldname</u> and <u>newname</u>, represent the old module-name and the new module-name, respectively, and are separated by a comma. <u>oldname</u> and <u>newname</u> are one to eight characters, the first of which must not be an asterisk.

Any number of RENAMR control statements can be used for the relocatable library within a single run.

For the rename function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

RENAMS

The RENAMS control statement is used to change the name of a book in the source statement library to another name. If the new name is already in the directory or an old name is not in the directory, an error message is issued. On a valid pair of operands, the new name simply replaces the old name in the directory; the version and modification levels are not changed. In either case, a check is then made for more operands on the card. As soon as the statement is processed, the system recognizes only the new book name. The RENAMS statement is in the following format.

```
RENAMS sublib.oldname,sublib.newname
   [,sublib.oldname,sublib.newname,...]
```

The operation field contains RENAMS. The qualifier <u>sublib</u> in the operand field represents the sublibrary containing the book to be renamed and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The <u>sublib</u> qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the sublibrary.

oldname and newname represent the old book name and the new book name. If newname is omitted, the name is assumed to be the same as for the oldname. The entries in the operand field must be separated by commas. The names in the operand field can be of any length; however, only a maximum of the first eight characters is used by the system to locate and rename the book.

The DOS/VS assemblers flag any reference to a macro in the source statement library as an UNDEFINED OPERATION CODE if the cataloged name of the macro is not identical to the operation code in the macro prototype statement. The assemblers locate macros in the source statement library by the cataloged name, but thereafter use the operation code of the macro prototype statement for identification.

Any number of RENAMS control statements can be used for the source statement library within a single run.

For the rename function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

RENAMP

The RENAMP control statement is used to change the name of a procedure in the procedure library to another name. If the new name is already in the directory or an old name is not in the directory, an error message is issued. On a valid pair of operands, the new name simply replaces the old name in the directory; the version and modification levels are not changed. In either case, a check is then made for more operands on the card. As soon as the statement is processed, the system recognizes only the new module name. The RENAMP statement is in the following format:

```
RENAMP
oldname,newname[,oldname,newname,...]
```

oldname, newname represent the old procedure name followed by the new procedure name. The procedure names are from one to eight alphameric characters, the first of which must be alphabetic or $. They must not be ALL.

Any number of RENAMP control statements can be used for the procedure library within a single run.

For the rename function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

CONDENSE

The condense function eliminates vacancies, resulting from delete or catalog functions. The condense function is used when a number of vacancies have accumulated within the library. Condense can be requested via one of two control statements:

CONDS -- condenses any or all of the libraries when issued

CONDL -- condenses any or all of the libraries automatically when a specified limit is reached.

CONDS

The CONDS control statement, in the following format, is used to condense the core image library:

```
CONDS CL
```

The operation field contains CONDS. The operand field contains CL. The relocatable library, the source statement library, and/or the procedure library can also be condensed in this run. If this is desired, the entry RL (for the relocatable library), SL (for the source statement library), and PL (for the procedure library) can appear in the operand field. Multiple entries in the operand field are separated by commas.

The CONDS statement, in the following format, is used to condense the relocatable library:

```
CONDS RL
```

The CONDS statement, in the following format, is used to condense the source statement library:

```
CONDS SL
```

The CONDS statement, in the following format, is used to condense the procedure library:

```
CONDS PL
```

For the condense function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, SYSLOG must not be assigned to a printer.

CONDL

The condense maintenance function can be performed automatically at the end of a catalog or delete maintenance function under the control of an installation specified parameter. The parameter is stored in the system directory. It indicates that when the number of blocks available in the corresponding library is less than the number specified by the parameter, and at least one directory entry is deleted, the condense function is performed for that library. The system interrogates the parameter at the completion of each maintenance function for the library. If a condense function is to be performed, a message appears on SYSLOG to inform the operator that the library is to be condensed. If a private core image library is assigned to more than one partition, the condense is not performed. If multiprogramming is in progress, the auto-condense is ignored for any library except a private core image library. Also, if condense (CONDS) is specified when MAINT is executed and a private core image library is assigned to a foreground partition, the condense is not performed.

The CONDL control statement (as opposed to the CONDS control statement for a user-specified condense function) informs the MAINT librarian program that a parameter to specify an automatic condense is to be set. The CONDL control statement is in the following format.

```
CONDL lib=nnnnn[,lib=nnnnn,...]
```

In the operand field, the entry lib is CL for the core image library, RL for the relocatable library, SL for the source statement library, and PL for the procedure library. The entry nnnnn represents the number of blocks specified for the specific library and is from one to five decimal digits. The maximum value of nnnnn is 65536. Each track of the core image library contains 3 blocks on the 2311 (private core image library only), 6 blocks on the 2314/2319, 11 blocks on the 3330/3333, or 7 blocks on the 3340. Each track of the relocatable library contains 16 blocks on the 2314/2319, 28 blocks on the 3330/3333, or 17 blocks on the 3340. Each track of the source statement library contains 27 blocks on the 2314/2319, 44 blocks on the 3330/3333, or 26 blocks on the 3340. Each track of the procedure library contains 40 blocks on the 2314/2319, 61 blocks on the 3330/3333, or 34 blocks on the 3340.

If 0 (zero) is specified for nnnnn, an automatic condense is not performed for the specific library. If the number of blocks specified exceeds the number of blocks allocated for the library, a condense is performed each time deleted blocks appear in the library at the end of a maintenance function. When the system is copied onto another pack, the condense limit on SYSRES is set to zero.

The automatic condense of the core image library is bypassed when a new supervisor is cataloged.

The condense limits are displayed with the system status on a DSERV and at the end of a maintenance job.

The control statement input to establish a value for an automatic condense is the same as that for a user-specified condense. See the section CONDS for a description of the control statement input.

REALLOCATE

The reallocation function redefines the number of tracks and cylinders allotted to a disk-resident system.

The EXEC control statement required to perform a reallocation function is in the following format:

```
// EXEC MAINT
```

Associated with the EXEC statement for the reallocation function is the ALLOC control statement. The ALLOC control statement is in the following format:

```
ALLOC CL=cylin(tracks),RL=cylin(tracks),
      SL=cylin(tracks),PL=cylin(tracks)
```

The operation field contains ALLOC. The operand field contains a sequence of letter combinations and cylin(tracks). The library identifiers are as follows:

CL = core image library and directory
RL = relocatable library and directory
SL = source statement library and directory
PL = procedure library and directory.

All operands of the ALLOC statement are required. If a specific library is not present, an allocation of 0(0) must be supplied.

The entry, cylin, in the operand field refers to the number of cylinders that contain the specified library. The entry, track, is enclosed within parentheses and refers to the number of tracks that contain the specified library directory. The tracks allocated to the directory are contained in the cylinders allocated to the library. The keyword operands are separated by a comma.

For the reallocation function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, or a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

If message 4n44A comes up due to a change in the label set when reallocating SYSRES, type DELETE on SYSLOG. The job continues using new extents.

UPDATE

The update function updates properly identified statements within a book of a source statement library.

The UPDATE control statement has the following format:

```
UPDATE sublib.bookname,[s.book1],[v.m],[nn]
```

The operation field contains UPDATE.

sublib in the operand field represents the sublibrary that contains the book to be updated. It may be any of the characters A-Z, 0-9, $, @, or #.

bookname represents the book that is to be updated in the sublibrary.

s.book1 in the operand field provides a temporary update option. The old book is renamed s.book1 and the updated book is named sublib.bookname. s indicates the sublibrary that contains the old, renamed book. It may be one of the characters A-Z, 0-9, $, @, or #. If this operand is not specified, the old book is deleted.

v.m represents the change level of the book to be updated. v may be any decimal number from 0-127. m may be any decimal number from 0-255. This operand must be present if change level verification is to be done, and it must correspond to the change level in the book's directory entry. If the change level is verified, the change level in the book's directory entry is increased by 1 for verification of the next update. If m is at its maximum value and an update is processed, m is reset to 0 and the value of v is increased by 1.

If both v and m are at their maximum values and an update is processed, both v and m are reset to zeros. If the directory entry specifies that change level verification is not required before updating, the change level operand in the statement is ignored. Use of the optional entry C in the CATALS control statement at the time the book is cataloged to the library determines if change level verification is required before updating.

nn in the operand field represents the resequencing status required for the update. nn may be a one- or two-character decimal number from 1-10, or it may be the characters FS or the word NO. If nn is a decimal number, it represents the increment that will be used in resequencing the statements in the book. For books supplied by IBM since Release 28, nn must be FS (fixed sequence). When nn is FS, no resequencing can be performed. If nn is NO, the statements will not be resequenced. If nn is not specified, the statements will be resequenced with an increment of 1. When a book is resequenced, the sequence number of the first statement is padded with zeros.

Note: See the section CATALS for additional information about sequence numbering books in a source statement library.

The UPDATE control statement is followed by ADD, DEL (delete), and/or REP (replace) control statements as required. Each ADD or REP statement is followed by source statements that are to be added to the book. The update section is terminated by an END statement. The ADD, DEL, REP, and END statements are identified as update control statements by a right parenthesis in the first position (column 1 in card format). The second position is blank. This is a variation from the general librarian control statement format, but it clearly identifies these control statements as part of the update function.

ADD Statement

The ADD statement is used for the addition of source statements to a book. The format is:

```
) ADD seq-no
```

ADD indicates that source statements following this statement are to be added to the book.

seq-no represents the sequence number of the statement in the book after which the new statements are to be added. It may be any decimal number from one to four characters in length (columns 77-80). If FS was specified in the UPDATE control statement, seq-no may be any decimal number from one to six characters in length (columns 73-78). (Columns 79-80 of IBM-supplied books contain the release number.) If FS was specified, also note that the seq-no of the first and only the first) ADD statement may be zero. In this case, the source statements that follow are added in front of the first source statements of the book (provided that their sequence numbers are lower than those of the first source statements in the book).

DEL Statement

The DEL statement causes the deletion of source statements from the book. The format is:

```
) DEL first-seq-no[,last-seq-no]
```

DEL indicates that statements are to be deleted from the book.

first-seq-no and last-seq-no represent the sequence numbers of the first and last statements of a section to be deleted. Each number may be any decimal number from one to four characters in length (columns 77-80). If FS was specified in the UPDATE control statement, each number may be any decimal number from one to six characters in length (columns 73-78). Note: The first-seq-no must not be zero. If last-seq-no is not specified, the statement represented by first-seq-no is the only statement deleted.

To delete an entire book, use the DELETS function.

REP Statement

The REP statement is used when replacement of source statements in a book is required. The format is:

```
) REP first-seq-no[,last-seq-no]
```

This indicates that source statements following the REP statement are to replace existing source statements in a book.

first-seq-no and last-seq-no represent the sequence numbers of the first and last statements of a section to be replaced. Each number may be a decimal number from one to four characters in length (columns 77-80). If FS was specified in the UPDATE control statement, each number may be any decimal number from one to six characters in length (columns 73-78). Note: The first-seq-no must not be zero. Any number of new statements can be added to a book when a section is replaced. (The number of statements added need not equal the number of statements being replaced.)

To replace an entire book, use the DELETS and CATALS functions.

If FS was not specified in the UPDATE control statement, sequence number 9999 is the highest number acceptable for a statement to be updated. If the book is so large that statement sequence numbers have wrapped around to zeros, it is not possible to update.

If FS was specified in the UPDATE control statement, 999999 is the maximum. In this case, 999999 need not be an actual sequence number. Specifying 999999 as the last-seq-no in)DEL or) REP cards indicates that the source statements beginning with the first-seq-no and up to and including the last source statement of the book are to be deleted or replaced.

If FS was specified in the UPDATE control statement, any source statement following the) ADD or) REP control statement should have the following format:

Columns

1-72 contain any user's source statement.

73-78 must contain a sequence number which must be greater than the sequence number after which it is inserted and less than the subsequent sequence number.

79-80 may contain any character including blanks.

Note: If source code following) ADD or) REP contains blanks in columns 73-78, the sequence number of that source code will be defaulted to the previous sequence number plus one, and columns 79/80 will be filled in by **.

If FS was not specified in the UPDATE control statement, any source statement following the) ADD or) REP control statement should have the following format:

Columns

1-72 contain any user's source statement.

73-76 contain a program identifier.

77-80 may contain a sequence number. If used, each number must be greater than the sequence number after which it is inserted, and less than the subsequent sequence number.

Note: See the section CATALS for additional information about sequence numbering books in a source statement library.

END Statement

This statement indicates the end of an update to a book. The format is:

```
) END [v.m[,C]]
```

END indicates the end of updates to a book.

The v.m operand provides another means of explicitly setting the change level of a book in the library. (The other way is through the use of the v.m operand in the CATALS statement.) v may be any decimal number from 0-127. m may be any decimal number from 0-255.

C indicates that change level verification is required before any subsequent updates to this book.

If v.m is specified and C is omitted, the book does not require change level verification before a subsequent update. This feature removes a previously specified verification requirement for a particular book.

If both optional operands are omitted, and change level verification was required for this update, the change level in the book's directory entry is increased, as the result of the update, and the verification requirement remains unchanged.

UPDATE FUNCTION INVALID OPERAND DEFAULTS

UPDATE Statement

1. If the first or second operand is invalid, the statement is flagged, the book is not updated, and the remaining control statements are checked for validity.

2. If change level verification is required and the wrong change level is specified, the statement is flagged, the book is not updated, and the remaining control statements are checked for validity.

3. If the resequencing operand is invalid, resequencing is done in increments of 1. The job will be canceled if FS was required.

ADD, DEL, or REP Statements:

1. If there is an invalid operation or operand in an ADD, DEL, or REP statement, the statement is flagged, the book is not updated, and the remaining control statements are checked for validity. All options of the UPDATE and END statements are ignored.

2. The second operand must be greater than the first operand in a DEL or REP statement. If not, the statement is considered invalid, it is flagged, the book is not updated, and the remaining control statements are checked for validity. If a second operand is present on an ADD statement, it is flagged as an invalid operand and ignored. The book is updated and normal processing continues. All options of the UPDATE and END statements are ignored.

3. All updating to a book between an UPDATE statement and an END statement must be in ascending sequential order of statement sequence numbers. The first operand of a DEL or REP statement must be greater than the last operand of the preceding control statement. The operand of an ADD statement must be equal to or greater than the last operand of the preceding control statement. Consecutive ADD statements must not have the same operand. If these conditions are not met, the default is the same as for items 1 and 2.

END Statement: If the first operand of the END statement is invalid, the statement is flagged, both operands are ignored, and the book is updated as though no operands were present. If the second operand is invalid, the statement is flagged, the operand is ignored, and the book is updated as though the second operand was not specified.

Cut-of-Sequence Updates: If FS was specified in the UPDATE control statement, the source statements to be added to a book must be in sequence; otherwise, the book is not updated and the remaining control statements are checked for validity. If the source statements to be added do not contain sequence numbers, the sequence number will be defaulted to the previous sequence number plus one, and columns 79-80 will be filled in by **.

If FS was not specified in the UPDATE control statement, and if the source statements to be added to a book are not in sequence, or do not contain sequence numbers, the book is updated and a message indicating the error appears following the END statement. If the resequencing option has been specified in the UPDATE statement, the book is sequenced by the specified value and subsequent updating is possible. If the resequencing option is not specified, the book is resequenced in increments of 1 and subsequent updating is possible. If the resequencing option NO is specified, the updated book is not in sequence and subsequent updating may not be possible.

For the update function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

<u>UPDATE Function - Activity Log</u>: The UPDATE function provides a log, on SYSLST, of all update activity to books in the source statement library. The log indicates the operation performed and all operands of the update control statements. It also shows the contents of the source statements affected, including the old prograr identification and sequence number, the new program identification and sequence number, and the update activity involved. Figures 16 and 17 show examples of an UPDATE job stream and an UPDATE activity log, respectively.

If no resequencing has been specified, only the old ID and sequence numbers of the statements involved are indicated.

```
// JOB UPDATE
// EXEC MAINT
  UPDATE A.TESTCASE
) REP 0032
OPTNFND    CLC     0(1,REGG),BLANKS
) REP 0039,0044
OPTNCHK    CLC     0(5,REGG),=C'PUNCH'
           BE      PUNCH
           CLC     0(5,REGG),=C'DSPCH'
) ADD 0052
PUNCH      MVI     SWITCH,C'X'
           B       CHKOPND
) DEL 0134,0135
) END
/&
```

Figure 16. Example of an UPDATE Job Stream

```
UPDATE   A.TESTCASE

 ) REP 0032
   OPTNFND  CLI   0(REGG),C' '                                A4530032
   OPTNFND  CLC   0(1,REGG),BLANKS                            A4530032   A4530027    REPLACEMENT

 ) REP 0039,0044
   OPTNCHK  CLC   0(5,REGG),PUNCHOP                           A4530039
            BNE   CHKDSP                                      A4530040
            MVI   SWITCH,C'X'                                 A4530041
            B     CHKOPND                                     A4530042
   CHKDSP   CLC   0(5,REGG),DSPCHOP                           A4530043
            BNE   MSGLOG                                      A4530044
   OPTNCHK  CLC   0(5,REGG),=C'PUNCH '                        A4530039   A4530034    REPLACEMENT
            BE    PUNCH                                       A4530040   A4530035    REPLACEMENT
            CLC   0(5,REGG),=C'DSPCH '                        A4530041   A4530036    REPLACEMENT

 ) ADD 0052
            B     M6GLOG                                      A4530052   A4530044    RESEQUENCED
   PUNCH    MVI   SWITCH,C'X'                                 A4530053   A4530045    ADD
            B     CHKOPND                                     A4530054   A4530046    ADD

 ) DEL 0134,0135
   DSPCHOP  DC    C'DSPCH '                                   A4530134               DELETE
   PUNCHOP  DC    C'PUNCH '                                   A4530135               DELETE

 ) END
```

Figure 17. Example of an UPDATE Activity Log

Example I

```
col. 1                                                  col. 73-80

// EXEC MAINT
   UPDATE Z.TESTBOOK,Z.SAVEBOOK,,FS
) ADD 24310
     CLC    SSEXTNT(4),ZERO            SSL ALLOCATED  024313AS
     BE     IGNORE                     NO             024316AS
) REP 25430
     BCT    WRKREG2,LOOP               LOOP TWICE     025430
) DEL 26210,26230
) END
/*
```

EXAMPLE OF AN UPDATE JOB STREAM FOR IBM-SUPPLIED BOOKS

A book in the sublibrary Z named TESTBOOK has to be updated temporarily. The old book is to be renamed: SAVEBOOK. Two statements are to be added after the source statement with sequence number (columns 73-78): 024310. One statement with sequence number 025430 is to be replaced. Three subsequent statements with sequence numbers: 26210, 26220, 26230 are to be deleted. The control statements are shown in Example I.

Copy Program

This section describes the control statements that can be submitted to the CORGZ program, which performs the following operations, individually or in combination:

1. Defines and/or creates a new system pack.
2. Defines and/or creates private libraries.
3. Transfers phases, modules, or books between any assigned libraries.

To invoke the CORGZ program, use job control statement:

// EXEC CORGZ

Associated with the EXEC statement are three independent copy control statements and five COPY statements. The three independent copy control statements are:

1. ALLOC statement. Defines the libraries on a new system resident pack. If this statement is omitted, or if any of the libraries is not allocated in the statement, the job is canceled. If no assignments for a particular library are to be made, the assignment should nevertheless be included with a zero allocation. A zero allocation is not possible for the system core image library. The directory allocation for the CIL is at least two tracks: one directory for cataloged phases and one directory for linked phases. (The first two tracks of the system residence file should be taken into account for the number of cylinders allocated to the SCIL. These two tracks are used by the system for IPL information and a system workarea.)

   The format of the ALLOC statement is identical to the ALLOC statement described earlier in the section Reallocate.

   To use the reallocated or newly created pack as system residence file, you must re-IPL this pack again.

2. NEWVOL statement (see the section Creation of Private Libraries). Defines private libraries.

3. MERGE statement (see the section Merge Function). Transfers data between libraries that were defined, or defined and created previously.

The following functions are performed automatically by the CORGZ program, when the ALLOC statement is used:

- All programs essential to a minimum system are copied. These programs are all logical and physical transients, IPL, supervisor, job control, and linkage editor.
- The partition and system standard labels are copied from the SYSRES label cylinder to the label cylinder on SYS002.
- Cylinder 0, track 0, records 1 and 2, and cylinder 0, track 1 are built.

COPY, ALL SYSTEM LIBRARIES

The COPY control statement copies the complete system, but can only be used when SYSRES contains all four libraries. The information copied into the new pack includes cylinder 0 (track 0, records 1 and 2, and track 1), the four libraries, the label cylinder, and the VTOC cylinder. It is in the following format COPY ALL

This statement is valid only when preceded by an ALLOC statement.

COPY, CORE IMAGE LIBRARY

The COPYC control statement is used to specify the phases or programs in the core image library that are to be copied. It is in one of the following formats.

```
COPYC phase1[,phase2,...]

COPYC prog1.ALL[,prog2.ALL,...]

COPYC ALL
```

The first format is used when specific phases are to be copied. The entry in the operation field is COPYC. The entry, *phase*, in the operand field represents the name(s) of the phase(s) to be copied. Entries in the operand field must be separated by commas.

The second format is used when specific programs are to be copied. The entry in the operation field is COPYC. The entry, *prog.ALL*, in the operand field represents the name of the program to be copied. *prog* is the first four characters of the program name. (All phases within a program have the same first four characters.) *prog* is followed by a period and ALL. Entries in the operand field must be separated by commas.

The third format is used to copy the complete core image library. The entry in the operation field is COPYC. The entry in the operand field is ALL.

COPY, RELOCATABLE LIBRARY

The COPYR control statement is used to specify the modules in the relocatable library that are to be copied. It is in one of the following formats.

```
COPYR module1[,module2,...]

COPYR prog1.ALL[,prog2.ALL,...]

COPYR ALL
```

The first format is used when specific modules are to be copied. The entry in the operand field is COPYR. The entry, *module*, in the operand field represents the name(s) of the module(s) to be copied. Entries in the operand field must be separated by commas.

The second format is used when specific programs are to be copied. The entry in the operation field is COPYR. The entry, *prog*, in the operand field represents the name of the program to be copied. *prog* is the first three characters of the program name. (All modules within an IBM-supplied program have the same first three characters, such as IJB for the supervisor and IJK for PL/I). *prog* is followed by a period and ALL. Entries in the operand field must be separated by commas.

The third format is used to copy the complete relocatable library. The entry in the operation field is COPYR. The entry in the operand field is ALL.

COPY, SOURCE STATEMENT LIBRARY

The COPYS control statement is used to specify the books in the source statement library that are to be copied. It is in one of the following formats.

```
COPYS sublib.book1[,sublib.book2,...]

COPYS sublib1.ALL[,sublib2.ALL,...]

COPYS ALL
```

The first format is used when specific books are to be copied. The entry in the operation field is COPYS. The qualifier *sublib* in the operand field represents the name of the sublibrary containing the book and can be an alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The *sublib* qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the book.

book represents the name(s) of the book(s) to be copied.

The second format is used when an entire sublibrary is to be copied. The entry in the operation field is COPYS. The entry, *sublib*, in the operand field represents the name of the sublibrary to be copied and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The *sublib* qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the sublibrary. The qualifier sublib is followed by a period and ALL.

The third format is used to copy the complete source statement library. The entry in the operation field is COPYS. The entry in the operand field is ALL.

COPY, PROCEDURE LIBRARY

The COPYP control statement is used to specify the procedures in the procedure library that are to be copied. It has the following format:

```
COPYP procedurename[,procedurename,...]
      ALL
```

procedurename specifies the name of the procedure to be copied. If more than one procedure is to be copied, the names must be separated by commas.

ALL specifies that the entire procedure library is to be copied.

COPY, IPL

The COPYI control statement is used to transfer the IPL Retrieval Program ($$A$IPL2) between the SYSRES and SYS002 (RES and NRS) volumes. It is in the following format.

```
COPYI $$A$IPL2
```

COPY CONSIDERATIONS

Each library that is to be selectively copied requires a separate group of control statements. If more than one copy control statement for a particular library (COPYC, COPYR, COPYS, or COPYP) is submitted to the CORGZ program in combination with copy statements for a different library, the statements must be grouped per library (for instance: first all COPYC statements, then all COPYR statements, and so on).

A COPY ALL statement must neither be preceded nor followed by any other COPY statement. Any number of elements of a particular library can be specified in one control statement. Continuation statements are not valid. All entries in the operand field must be separated by commas.

When executing the copy disk (CORGZ) function, a file must be defined for IJSYSRS on SYS002 via DLBL and EXTENT control statements. The filename on the DLBL statement must be IJSYSRS. The file identification portion of the DLBL statement can be as shown in the example of the copy function.

The lower extent for this file must be cylinder zero, track one, and the upper extent must include the label cylinder. The label cylinder is one cylinder reserved for label information, and is located on the cylinder immediately following the last library of the system. The total allocation must include cylinder 0, all defined system libraries, and the label cylinder.

Figure 18 shows an example of a valid job setup for the copy function on a 2316 disk pack.

For the copy function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYS002 must be assigned to a disk unit. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

CREATION OF PRIVATE LIBRARIES

The COPY program can be used to create private libraries. SYS003 must be assigned if a private core image library is required. SYSRLB must be assigned if a private relocatable library is required. SYSSLB must be assigned if a private source statement library is required.

```
// JOB    COPY SYSRES TO NRES
// ASSGN  SYS002,X'191'
// DLBL   IJSYSRS,'DOS/VS SYSTEM RESIDENCE FILE',99/365,SD
// EXTENT SYS002,111111,1,000,00001,02519
// EXEC   CORGZ
   ALLOC  CL=60(10),RL=30(10),SL=30(10),PL=5(5)
   COPY   ALL
/*
/&
```

Figure 18. Example of a Valid Job Setup for the Copy Function on a 2316 Disk Pack

Private libraries can reside on the SYSRES pack (outside the SYSRES extent) or on separate disk packs which (except for a private core image library) must be of the same device type as the SYSRES pack.

When creating a private library using CORGZ, a file must be defined for IJSYSPC on SYS003 and/or IJSYSRL on SYSRLB and/or IJSYSSL on SYSSLB. IJSYSPC, IJSYSRL, and IJSYSSL are the respective file names used in the creation of private core image, relocatable, and source statement libraries.

Creation of a private library is requested by the NEWVOL control statement. Its format follows.

```
NEWVOL id=cylin(track)[,id=cylin(track)]
```

id — Indicates the specific library and directory to be created and can be:

CL for a private core image library and directory

RL for a private relocatable library and directory

SL for a private source statement library and directory.

cylin — Indicates the number of cylinders to contain the specified library.

track — Indicates the number of tracks to contain the specified library directory. The tracks allocated to the directory are contained within the cylinders allocated to the library. The difference between the number of cylinders for the library and the number of tracks for the library directory has to be at least 5 tracks.

The COPY program also provides the ability to copy all or part of the system core image and/or relocatable and/or source statement library into its respective private library. If this facility is to be used, it must be employed in the same job step in which the private library is created. This is done by inserting COPYC and/or COPYR and/or COPYS statements immediately behind the NEWVOL statement(s) in the job stream.

To define and create a private library from an existing private library, the MERGE statement must be used between the NEWVOL and the COPYR and/or COPYS statements (This does not apply for the COPYC statement, since the private core image library cannot be defined and created in one single job step). Below is an example of the sequence of steps.

```
// EXEC CORGZ
   NEWVOL RL=10(2),SL=10(1)
   MERGE PRV,PRV
   COPYR ALL
   COPYS ALL
/*
/&
```

The following precautions should be observed.

1. When a NEWVOL statement and a COPYC and/or a COPYR and/or a COPYS statement are both present, the NEWVOL statement must precede the COPYC, COPYR or COPYS statement.

2. The creation of a new system residence file and creation of private libraries cannot both be done in the same job step.

3. For each job, label cards for the private libraries containing the same information as at creation time must be submitted. Note: Private libraries are not required to begin on a cylinder boundary.

MERGE FUNCTION

The MERGE function copies the contents of one core image, relocatable, source statement, or procedure library to another core image, relocatable, source statement, or procedure library, respectively. If the phase, module, book, or procedure being copied already exists in the library being updated, the resident phase, module, book, or procedure becomes inaccessible because its directory entry is deleted. Any phase, module, book, or procedure being copied is added to the library at the next available entry point. Its directory entry is added at the directory's next available entry point.

The control statement indicates the type of library (resident or private) involved and the direction in which the data will move. It has the following format:

```
MERGE from,to
```

The operand field entry from represents the file from which data will be copied. from can be one of these:

RES — System residence file on SYSRES.

NRS — Modified, or duplicate system residence file on SYS002 (New ReSidence).

PRV Private relocatable library on SYS001 and/or a private source statement library on SYS000 and/or a private core image library on SYS003.

The operand field entry to indicates the file to which library data will be transferred. to can be one of these:

RES System residence file on SYSRES.

NRS Modified, or duplicate system residence file on SYS002 (New ReSidence).

PRV Private relocatable library on SYSRLB and/or private source statement library on SYSSLB and/or a private core image library on SYSCLB.

The MERGE statement is followed by appropriate copy statements (COPYC, COPYR, COPYS, COPYP) that indicate the phases, modules, books, or procedures to be transferred.

All copy statements following a MERGE statement apply to that function until another MERGE, NEWVOL, or ALLOC statement is encountered.

Notes:

1. If the COPYC ALL statement is used, the supervisor and transient phases are also transferred to the receiving system resident file or private core image library. The supervisor and transients previously contained on the receiving SYSRES or SYSCLB disk pack are deleted. No indication of this deletion is given, and it is the user's responsibility to ensure that the receiving system is able to continue operating.

2. If COPY ALL or COPYI $$A$IPL2 is specified, the IPL bootstrap phase at cylinder 0, track 1, record 5 of SYSRES is copied.

Core Image Library

Selected phases, or the entire library, can be transferred (in either direction) between:

1. The core image library of the system residence file on SYSRES and the core image library of another system residence file on SYS002.

2. A private core image library and the core image library of the system residence file on SYSRES.

3. A private core image library and the core image library of a system residence file on SYS002.

4. Two private core image libraries.

You must ascertain that the receiving system has the ability to execute the phase(s) being copied.
If the receiving file is the system residence file on SYSRES, CORGZ must be executed in the background partition.

Relocatable Library

Selected modules, or the entire library, can be transferred (in either direction) between:

1. The relocatable library of the system residence file on SYSRES and the relocatable library of another system residence file on SYS002.

2. A private relocatable library and the relocatable library of the system residence file on SYSRES.

3. A private relocatable library and the relocatable library of a system residence file on SYS002.

4. Two private relocatable libraries.

Source Statement Library

Selected books or the entire library, can be transferred (in either direction) between:

1. The source statement library of the system residence file on SYSRES and the source statement library of another system residence file on SYS002.

2. A private source statement library and the source statement library of the system residence file on SYSRES.

3. A private source statement library and the source statement library of a system residence file on SYS002.

4. Two private source statement libraries.

Procedure Library

Selected procedures, or the entire procedure library, can be transferred (in either direction) between the procedure library of the system residence file on SYSRES and the procedure library of another system residence file on SYS002.

MERGE Considerations

File definitions (through DLBL and EXTENT statements) must be made before the MERGE control statement is used. When defining files, remember:

1. When merging to, or from, a modified or duplicate system residence file, the modified or duplicate file name must be IJSYSRS, the logical unit must be SYS002, and the file ID must be identical to the ID supplied when the file was created.

2. When merging to a private relocatable library file, the file name must be IJSYSRL, the logical unit must be SYSRLB, and the file ID must be identical to the ID supplied when the file was created.

3. When merging from a private relocatable library file, the file name must be IJSYSPR, the logical unit must be SYS001, and the file ID must be identical to the ID supplied when the file was created.

4. When merging to a private source statement library file, the file name must be IJSYSSL, the logical unit must be SYSSLB, and the file ID must be identical to the ID supplied when the file was created.

5. When merging from a private source statement library file, the file name must be IJSYSPS, the logical unit must be SYS000 and the file ID must be identical to the ID supplied when the file was created.

6. When merging to a private core image library file, the file name must be IJSYSCL, the logical unit must be SYSCLB, and the file ID must be identical to the ID supplied when the file was created.

7, When merging from a private core image library file, the file name must be IJSYSPC, the logical unit must be SYS003 and the file ID must be identical to the ID supplied when the file was created.

Figure 19 shows the file name, logical unit, and direction of transfer for each of the MERGE operations. Any combination of the indicated operations can be performed in one job step.

Diagnostic messages for erroneous assignments, file definitions, etc, are provided on SYSLST. Figure 20 is an example of a job set up to use the MERGE function on a 2316 disk pack.

		1	2	3	4	5	6	7
File Name Logical Unit	IJSYSRS SYSRES	IJSYSRS SYS002	IJSYSRL SYSRLB	IJSYSPR SYS001	IJSYSSL SYSSLB	IJSYSPS SYS000	IJSYSCL SYSCLB	IJSYSPC SYS003
Merge RES to NRS	from	to						
Merge NRS to RES	to	from						
Merge RES to PRV	from		to		to		to	
Merge NRS to PRV		from	to		to		to	
Merge PRV to RES	to			from		from		from
Merge PRV to NRS		to		from		from		from
Merge PRV to PRV			to	from	to	from	to	from

Figure 19. Direction of Transfer for Merge Operations

```
        Assume two disk drives with addresses of 190 and 191.

           // JOB EXAMPLE
           // ASSGN SYSRLB,X'191'
           // ASSGN SYSSLB,X'191'
           // ASSGN SYS003,X'191'
           // ASSGN SYS000,X'190'
           // ASSGN SYS001,X'190'
           // ASSGN SYS002,X'191'

 Note 1    // DLBL IJSYSRL,'PRIVATE RL',99/365
           // EXTENT SYSRLB,111111,1,0,1500,200

 Note 2    // DLBL IJSYSSL,'PRIVATE SL',99/365
           // EXTENT SYSSLB,111111,1,0,1700,200

 Note 3    // DLBL IJSYSPC,'PRIVATE CIL',99/365
           // EXTENT SYS003,111111,1,0,1900,200

 Note 4    // DLBL IJSYSPR,'PRIVATE RL TEST',99/365
           // EXTENT SYS001,111111,1,0,1300,200

 Note 5    // DLBL IJSYSPS,'PRIVATE SL TEST',99/365
           // EXTENT SYS000,111111,1,0,1500,200

 Note 6    // DLBL IJSYSRS,'SYSTEM RESIDENCE',99/365
           // EXTENT SYS002,111111,1,0,1,359

           // EXEC CORGZ
              NEWVOL RL=10(2),SL=10(2),CL=10(15)
 Note 7       COPYR ALL
              COPYS ALL
              COPYC ALL
              MERGE PRV,PRV
 Note 8       COPYR ALL
              COPYS ALL
              MERGE NRS,PRV
 Note 9       COPYR ALL
              COPYS ALL
           /*
           // ASSGN SYS003,X'190'
           // ASSGN SYS002,X'191'

 Note 10   // DLBL IJSYSCL,'PRIVATE CIL',99/365
           // EXTENT SYSCLB,111111,1,0,1900,200

 Note 11   ASSGN SYSCLB,X'191'

 Note 12   // DLBL IJSYSPC,'PRIVATE CL TEST',99/365
           // EXTENT SYS003,111111,1,0,1500,200

 Note 13   // DLBL IJSYSRS,'SYSTEM RESIDENCE',99/365
           // EXTENT SYS002,111111,1,0,1,359

           // EXEC CORGZ

 Note 14      MERGE PRV,PRV
              COPYC ALL

 Note 15      MERGE NRS,PRV
              COPYC ALL

           /&
```

Figure 20. Example of Job Set Up to Use the MERGE Function on a 2316 Disk Pack (Part 1 of 2)

Note 1: File definition statements for a private relocatable library file to be created and updated.

Note 2: File definition statements for a private source statement library file to be created and updated.

Note 3: File definition statements for a private core image library file to be created.

Note 4: File definition statements for a private relocatable library file from which modules to be copied.

Note 5: File definition statements for a private source statement library file from which books are to be copied.

Note 6: File definition statements for a modified, or duplicate system residence file from which modules and books are to be copied.

Note 7: Creates private core image, relocatable, and source statement libraries on SYS003, SYSRLB, and SYSSLB, and copies the core image, relocatable, and source statement libraries from the system residence file on SYSRES into them.

Note 8: Merges all modules and books from private relocatable and source statement libraries on SYS001 and SYS000 into the appropriate private libraries created on SYSRLB and SYSSLB.

Note 9: Merges all modules and books from the relocatable and source statement libraries of a modified, or duplicate system residence file on SYS002 into private libraries created on SYSRLB and SYSSLB.

Note 10: File definition statements for the private core image library file just created and to be updated.

Note 11: In order to merge to the private core image library just created, assign it to SYSCLB. The from file must be assigned to SYS003 and is a previously created private core image library. See MERGE Considerations for additional information.

Note 12: File definition statements for a private core image library from which phases are to be copied.

Note 13: File definition statements for a modified, or duplicate system residence file from which the phases of the core image library are to be copied.

Note 14: Merges all phases from the private core image library on SYS003 into the newly created private core image library on SYSCLB.

Note 15: Merges all phases from the core image library of a modified, or duplicate system residence file on SYS002 into the newly created private core image library on SYSCLB.

Figure 20. Example of Job Set Up to Use the MERGE Function on a 2316 Disk Pack (Part 2 of 2)

Service Programs

This section describes the control statements that can be submitted to one of the programs that service the libraries. There are six service programs:

- DSERV -- to display the directories of each of the libraries.
- CSERV -- to display and/or punch phases from the core image library.
- RSERV -- to display and/or punch modules from the relocatable library.
- SSERV -- to display and/or punch books from the source statement library.
- ESERV -- to de-edit, display and/or punch, verify, and update edited assembler macros from the source statement library.
- PSERV -- to display and/or punch procedures from the procedure library.

The following symbolic units must be assigned to perform service functions on a private library: SYSCLB for a private core image library, SYSRLB for a private relocatable library, and SYSSLB for a private source statement library. A private library must be unassigned to perform service functions on the corresponding system library.

DSERV, DIRECTORY SERVICE

This section describes the display of the directories. The copy function for the directories is discussed in the section Copy Program.

The format of the display depends upon the operation specified in the DSERV control statement(s).

Multiple displays of the same directory, either sorted or unsorted, may be obtained in the same job step. To do this, use a separate control statement for each desired display.

Operational Characteristics

If any private library is assigned, it is displayed in place of the corresponding system library. To display the system library, the corresponding private library must be unassigned.

The printed output of DSERV contains the status report of the system libraries and any assigned private libraries followed by the specified directory. Each printed directory is preceded by a header that contains the name of the directory in EBCDIC characters. All fields are headed by the title DEC (decimal) or HEX (hexadecimal). Generally, each page includes up to 96 directory entries printed in double-column format with 48 entries per column. However, the core image directory contains more information and is therefore printed in single column format with up to 48 entries per page. The additional information in the printout of the core image directory in response to a DSPLY(S) request indicates:

1. If the phase is SVA-eligible
2. If the phase is listed in the SDL
3. If the phase is contained in the SVA
4. Entry names contained in the SDL that have been established for future use, but do not as yet have phases attached to them. These entry names are listed as contained in the SDL only.

DSERV can be executed in any partition with the statement:

```
// EXEC DSERV
```

Except for a display of the core image directory, DSERV may require additional storage beyond the minimum of 64K available to the partition if a sorted display is specified. If the partition is not large enough to contain all the entries to be sorted, sorting is done in more than one pass, resulting in more than one sorted list of entries. If this is not satisfactory, increase the partition size.

If DSERV is to run in real mode, the REAL parameter must be included in the EXEC statement.

SYSLOG must not be assigned to a printer and SYSIPT and SYSLST must be assigned.

DSERV Control Statements

The control statements for DSERV are:

```
DSPLY     directory[,directory]
DSPLYS    [(phasename[,nn])]
```

Operation

DSPLY Displays the directory entries in the sequence that they appear in the directory.

DSPLYS Displays the directory entries sorted alphamerically.

The entries in the core image directory are always displayed in alphameric sequence (DSPLY CD and DSPLYS CD give the same result).

Operands

Directory: Can be one of the following:

TD $-phases in the core image directory.

CD The core image directory

RD The relocatable directory

SD The source statement directory

PD The procedure directory

ALL All the above (TD, CD, RD, SD, and PD) are specified.

Note: If CD (without any other operands) or ALL is specified, the version and modification level of CD entries is not displayed.

Phasename: If CD is specified, then the version and modification level of any single phase or group of phases can be displayed by using phasename. Specify the phasename desired. If it contains less than eight characters, the specification should either be padded out with blanks (COBOL) or one blank should be included immediately after the last character of the phasename (COBOL). In the example, COBOL, if many phases contain the first five characters, COBOL, then to display the version and modification level of all of these phases specify (COBOL) without any blanks.

nn: Specify (in decimal) the displacement into the phase where the location of the version and modification level is for one of the following:

- A version and modification level in the nonstandard position.
- The version and modification level of the phase(s) specified is higher than that of the DSERV in use.

If nn is not specified, the program assumes 12 bytes for IBM-supplied transients and 8 bytes for all other entries.

Notes:

1. Only one phase or group of phases can be specified per CD control statement. If more than one is specified, then the last specification is the only one processed.
2. Continuation cards are not valid.
3. A blank operand or no control statement results in a system status report only.
4. The output of the core image directory also indicates if a phase is SVA-eligible, if it has an entry in the SDL (system directory list), and if it has been loaded in the SVA.

CSERV, CORE IMAGE LIBRARY

To request a service function for the core image library (either system or private), use the following EXEC control statement:

```
// EXEC CSERV
```

One or more of the three service functions can be requested within a single run. Punched output is sequenced in columns 77 through 80. The first card punched for each phase is sequenced zero. Any number of phases within the core image library can be acted upon in this run.

Display

The display function produces a printout of a phase in the core image library. The printed output produced by the display function consists of a header and the phase. The printed header contains the phase name and the length of the phase in number of bytes.

The printed output of the phase contains a three-byte hexadecimal load address of the first byte in the line, followed by 48 bytes of text displayed in hexadecimal.

If the phase is relocatable, relocation information is printed following the text information.

The DSPLY control statement in one of the following formats is used to display phases in the core image library.

```
DSPLY phase1[,phase2,...]

DSPLY prog1.ALL[,prog2.ALL,...]

DSPLY ALL
```

The first format is used if only specific phases are to be displayed. The entry in the operation field is DSPLY. phase in the operand field represents the name of the phase to be displayed. If more than one phase is to be displayed, the phase names are separated by commas. Phase names must be from one to eight characters long.

The second format is used when an entire program is to be displayed. The entry in the operation field is DSPLY. In the operand field, prog refers to the first four characters of the phase names making up a program. (All phases of a multiphase program should have the same first four characters.) The four characters are followed by a period and ALL.

The third format is used if the entire core image library is to be displayed. The entry in the operation field is DSPLY. The entry in the operand field is ALL.

It is possible to use all three types of operands together in a single control statement.

For the display function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette. SYSLOG must not be assigned to a printer.

Punch

The punch function converts a phase in the core image library into punched-cards or into a card-image format.

Any number of phases in the core image library can be punched within a single run unless SYSPCH is assigned to a disk or tape unit, or to a 3540 diskette (for subsequent use as SYSIPT to a link edit job step). In the case of a 3540 diskette a maximum of 120 phases can be punched. The punched-card output is acceptable as input to the linkage editor for recataloging to the core image library. When non-relocatable phases are recataloged, they are loaded in a partition at the same address as when they were originally link-edited because their address(es) are absolute. Relocatable phases, however, are loaded at an address relative to the end of the supervisor (S + displacement).

The following cards are contained in the phase decks punched from the core image library.

1. PHASE card: contains the phase name and the beginning load address. If the phase that is punched is SVA-eligible, that is, if it has been cataloged as such, the SVA parameter is also punched in the PHASE card.

2. ESD card: SD type, contains the phase name, the length of the phase, and the beginning load address.

3. TXT cards: contain the loading address of the first byte in the card, the number of bytes of text punched in the card (usually 56, except for the last card), the identification number of the control section (always 0001) containing the text, and the actual text.

4. END card: contains the transfer address, and signifies the end of the phase.

If the phase is relocatable, RLD cards are also punched. These cards contain information that identifies those portions of the TXT cards that may need to be modified by the relocating loader.

To facilitate recataloging of the phase(s), a /* card will be the last card of any punched output of a CSERV job step.

The PUNCH control statement in one of the following formats is used to convert phases in the core image library to punched-card output.

```
PUNCH phase1[,phase2,...]

PUNCH prog1.ALL[,prog2.ALL,...]

PUNCH ALL
```

The first format is used if only specific phases are to be punched. The entry in the operation field is PUNCH. The entry in the operand field, phase, represents the name of the phase to be punched. If more than one phase is to be punched, the phase names are separated by commas. Phase names must be from one to eight characters long.

The second format is used when an entire program is to be punched. The entry in the operation field is PUNCH. In the operand field, prog refers to the first four characters of the phase names making up a program. (All phases of a multiphase program should have the same first four characters.) The four characters are followed by a period and ALL.

The third format is used if the entire core image library is to be punched. The entry in the operation field is PUNCH. The entry in the operand field is ALL.

When SYSPCH is assigned to a tape or disk unit, each card image is preceded by a stacker-select character.

For the punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSPCH must be assigned to a card punch, a tape unit, or a disk unit. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the module must follow each PUNCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

Display and Punch

The display-and-punch function combines the separate operations of the display and the punch functions.

The DSPCH control statement is used to convert phases in the core image library to printed and punched-card output. The DSPCH control statement is in one of the following formats.

```
DSPCH phase1[,phase2,...]

DSPCH prog1.ALL[,prog2.ALL,...]

DSPCH ALL
```

The first format is used if only specific phases are to be displayed and punched. The entry in the operation field is DSPCH. The entry in the operand field, phase, represents the name of the phase to be displayed and punched. If more than one phase is to be displayed and punched, the phase names are separated by commas. Phase names must be from one to eight characters long.

The second format is used when an entire program is to be displayed and punched. The entry in the operation field is DSPCH. In the operand field, prog refers to the first four characters of the names of the phases making up the program. (All phases of a multiphase program should have the same first four characters.) The four characters are followed by a period and ALL.

The third format is used if the entire core image library is to be displayed and punched. The entry in the operation field is DSPCH. The entry in the operand field is ALL.

When SYSPCH is assigned to a tape or disk unit, each card image is preceded by a stacker-select character.

For the display and punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, or a disk unit. SYSPCH must be assigned to a card punch, a tape unit, a disk unit, or a 3540 diskette. SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the module must follow each DSPCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

RSERV, RELOCATABLE LIBRARY

To request a service function for the relocatable library, use the following EXEC control statement:

```
// EXEC RSERV
```

One or more of the three service functions can be requested within a single run. Any number of modules within the relocatable library can be acted upon in this run. Punched output is sequenced in columns 77 through 80. The first card punched for each module is sequenced zero.

Display

The display function produces a printout of a module in the relocatable library.

The printed output produced by the display function consists of a header and the module.

Contained in the printed header is the module name and the number of records needed to contain the module.

The printed output of the module is represented by hexadecimal characters and EBCDIC, depending on the type of record and the information contained within the record.

The DSPLY control statement in one of the following formats is used to display modules in the relocatable library.

```
DSPLY module1[,module2,...]

DSPLY prog1.ALL[,prog2.ALL,...]

DSPLY ALI
```

The first format is used if only specific modules are to be displayed. The entry in the operation field is DSPLY. module in the operand field represents the name of the module to be displayed. If more than one module is to be displayed, the module names are separated by commas. Module names must be from one to eight characters long.

The second format is used when an entire program is to be displayed. The entry in the operation field is DSPLY. In the operand field, prog refers to the first three characters of the modules used to build the program. (All IBM-supplied modules in the relocatable library making up a program have the same first three characters, such as IPK for the assembler and ILA and ILB for COBOL.) The three characters are followed by a period and ALL.

The third format is used if the entire relocatable library is to be displayed. The entry in the operation field is DSPLY. The entry in the operand field is ALL.

For the display function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette. SYSLOG must not be assigned to a printer.

Punch

The punch function converts a module in the relocatable library into a punched-card output check.

The PUNCH control statement in one of the following formats is used to convert modules in the relocatable library to punched-card output.

```
PUNCH module1[,module2,...]

PUNCH prog1.ALL[,prog2.ALL,...]

PUNCH ALL
```

The first format is used if only specific modules are to be punched. The entry in the operation field is PUNCH. The entry in the operand field, module, represents the name of the module to be punched. If more than one module is to be punched, the module names are separated by commas. Module names must be from one to eight characters long.

The second format is used when an entire program is to be punched. The entry in the operation field is PUNCH. In the operand field, prog refers to the first three characters of the modules used to build the program. (All IBM-supplied modules in the relocatable library making up a program have the same first three characters, such as IPK for the assembler and ILA and ILB ror COBOL.) The three characters are followed by a period and ALL.

The third format is used if the entire relocatable library is to be punched. The entry in the operation field is PUNCH. The entry in the operand field is ALL.

When SYSPCH is assigned to a tape or disk unit, each card image is preceded by a stacker-select character.

For the punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSPCH must be assigned to a card punch, a tape unit, or a disk unit. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette, and SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the module must follow each PUNCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

Display and Punch

The display-and-punch function combines the separate operations of the display and the punch functions.

The DSPCH control statement is used to convert modules in the relocatable library to printed and punched-card output. The DSPCH control statement is in one of the following formats.

```
DSPCH module1[,module2,...]

DSPCH prog1.ALL[,prog2.ALL,...]

DSPCH ALL
```

The first format is used if only specific modules are to be displayed and punched. The entry in the operation field is DSPCH. The entry in the operand field, module, represents the name of the module to be displayed and punched. If more than one module is to be displayed and punched, the module names are separated by commas. Module names must be from one to eight characters long.

The second format is used when an entire program is to be displayed and punched. The entry in the operation field is DSPCH. In the operand field, prog refers to the first three characters of the modules used to build the program. (All IBM-supplied modules in the relocatable library making up a program have the same first three characters, such as IPK for the assembler and ILA and ILB for COBOL.) The three characters are followed by a period and ALL.

The third format is used if the entire relocatable library is to be displayed and punched. The entry in the operation field is DSPCH. The entry in the operand field is ALL.

When SYSPCH is assigned to a tape or disk unit, or a 3540 diskette, each card image is preceded by a stacker-select character.

For the display and punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette. SYSPCH must be assigned to a card punch, a tape unit, or a disk unit. SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the module must follow each DSPCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

SSERV, SOURCE STATEMENT LIBRARY

To request a service function for the source statement library, use the following job control statement:

```
// EXEC SSERV
```

One or more of the three service functions can be requested within a single run. Any number of books within the source statement library can be acted upon in this run.

Display

The display function produces a printout of a book in the source statement library.

Books are displayed in the card image format. Each book is preceded and followed by a BKEND statement.

The DSPLY control statement in one of the following formats is used to display books in the source statement library.

- DSPLY sublib.book1[,sublib.book2,...]

- DSPLY sublib1.ALL[,sublib2.ALL,...]

- DSPLY ALL

The first format is used if only specific books are to be displayed. The entry in the operation field is DSPLY. The qualifier sublib in the operand field represents the sublibrary containing the book to be displayed and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

book in the operand field represents the name of the book in the sublibrary to be displayed. If more than one book is to be displayed the entries must be separated by commas. If books to be displayed are in the same sublibrary, subsequent book names need not be qualified. (The librarian assumes that nonqualified books are in the last sublibrary specified. If a sublibrary is never specified, the librarian assumes the book is in the E sublibrary.) The names of the books in the operand field can be from one to eight characters in length. Continuation statements are not recognized.

The second format is used if an entire sublibrary is to be displayed. The entry in the operation field is DSPLY. The first entry in the operand field is the name of the sublibrary to be displayed. The qualifier sublib can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The sublib qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the sublibrary.

The second entry in the operand field is ALL. The two entries must be separated by a period.

The third format is used if the entire source statement library is to be displayed. The entry in the operation field is DSPLY. The entry in the operand field is ALL.

For the display function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette. SYSLOG must not be assigned to a printer.

Punch

The punch function converts a book in the source statement library into punched cards or into card-image format.

The PUNCH control statement in one of the following formats is used to convert books in the source statement library to punched-card output.

- PUNCH sub.book1[,sub.book2,...][,CMPRSD]

- PUNCH sub1.ALL[,sub2.ALL,...][,CMPRSD]

- PUNCH ALL[,CMPRSD]

The first format is used if only specific books are to be punched. The entry in the operation field is PUNCH. The qualifier sub in the operand field represents the sublibrary containing the book to be punched and can be alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

book in the operand field represents the name of the book in the sublibrary to be punched. The entry CMPRSD is used if the books are to be punched in the compressed form in which they are stored in the library. When this option is elected, the cards are punched in the first seventy-one columns. If more than one book is to be punched, the entries must be separated by commas. If books to be punched are in the same sublibrary, subsequent book names need not be qualified. (The librarian assumes that nonqualified books are in the last sublibrary specified. If a sublibrary is never specified, the librarian assumes the book is in the E sublibrary.) The names of the books in the operand field can be from one to eight characters long. Continuation statements are not recognized.

The second format is used if an entire sublibrary is to be punched. The entry in the operation field is PUNCH. The first entry in the operand field is the name of the sublibrary to be punched. The qualifier sub can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The sublib qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the sublibrary.

The second entry in the operand field is ALL. The entry CMPRSD is used if the books are to be punched in the compressed format. A /* statement is always punched at the end of the output. When SYSPCH is assigned to a tape unit or disk unit, each card image is preceded by a stacker-select character.

The third format is used if the entire source statement library is to be punched. The entry in the operation field is PUNCH. The entry in the operand field is ALL. The entry CMPRSD is used if the books are to be punched in the compressed format. If CMPRSD is specified, punched output is sequenced in columns 77-80. The first card punched for each module is zero. The compressed source code retains its original sequence numbers. A /* statement is always punched at the end of the output.

For the punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSPCH must be assigned to a card punch, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, or a disk unit, and SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the book must follow each PUNCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

Display and Punch

The display-and-punch function combines the separate operations of the display function and the punch function.

The DSPCH control statement in one of the following formats is used to convert books in the source statement library to printed and punched-card output.

- DSPCH sub.book1[,sub.book2,...][,CMPRSD]
- DSPCH sub1.ALL[sub2.ALL,...][,CMPRSD]
- DSPCH ALL[,CMPRSD]

The first format is used if only specific books are to be displayed and punched. The entry in the operation field is DSPCH. The qualifier sub in the operand field represents the sublibrary containing the book to be displayed and punched and can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

book in the operand field represents the name of the book in the sublibrary to be displayed and punched. The entry CMPRSD is used if the books are to be punched in the compressed format, but printed in the original card image format. If more than one book is to be displayed and punched, the entries must be separated by commas. If books to be displayed and punched are in the same sublibrary, subsequent book names need not be qualified. (The librarian assumes that nonqualified books are in the last sublibrary specified. If a sublibrary is never specified, the librarian assumes the book is in the E sublibrary.) The names of the books in the operand field can be from one to eight characters long. Continuation statements are not recognized. A /* statement is punched at the end of the output.

The second format is used if an entire sublibrary is to be displayed and punched. The entry in the operation field is DSPCH. The first entry in the operand field is the name of the sublibrary to be displayed and punched. The qualifier sub can be any alphameric character (0-9, A-Z, #, $, and @), representing source statement sublibraries.

The sublib qualifier is required. If omitted, the operand is flagged as invalid and no processing is done on the sublibrary.

The second entry in the operand field is ALL. The entry CMPRSD is used if the books are to be punched in the compressed format. When SYSPCH is assigned to a tape or disk unit, each card image is preceded by a stacker-select character.

The third format is used if the entire source statement library is to be displayed and punched. The entry in the operation field is DSPCH. The entry in the operand field is ALL. The entry CMPRSD is used if the books are to be punched in the compressed format.

For the display and punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit. SYSPCH must be assigned to a card punch, a tape unit, a disk unit, or a 3540 diskette. SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the book must follow each DSPCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

ESERV, SOURCE STATEMENT LIBRARY

The DOS/VS assembler uses two sublibraries of the source statement library: the macro (E) sublibrary and the copy (A) sublibrary.

To request a service or maintenance function for edited assembler macros in the E sublibrary, specify:

// EXEC ESERV

Input to ESERV is read from SYSIPT. The first card after the // EXEC ESERV statement should be GENEND or GENCATALS. If neither is present, GENCATALS is assumed. These cards, as shown below, must not start in column 1.

ƀGENENDƀ no operands

This causes ESERV to place an END and a /* card immediately after the de-edited macro on SYSPCH so that it can be used as SYSIPT for the assembler.

ƀGENCATALSƀ no operands

This causes a CATALS statement of the form A.bookname (or, if SUBLIB=DF was specified in an OPTION statement, of the form D.bookname) to precede and a /* card to follow each macro so that the resulting SYSPCH stream can be used as SYSIPT input to the MAINT program for cataloging the de-edited macros into the A sublibrary.

Display and Punch Functions

The following functions can be performed for one or more edited macros within a single ESERV run.

1. DSPLY De-edits and displays macro(s) on SYSLST.

2. PUNCH De-edits and punches macro(s) on SYSPCH.

3. DSPCH Combines 1 and 2.

The control statements must not start in column 1; they must have the following format:

$\left\{\begin{array}{l}\text{DSPLY}\\ \text{PUNCH}\\ \text{DSPCH}\end{array}\right\}$ sublib.book1[,sublib.book2...]

If the qualifier sublib is omitted, the E sublibrary is assumed. If the de-edited macro is to be updated, the appropriate control statements must follow immediately. (See 4-10 below.)

Verification and Update Functions

These functions can be performed for a single de-edited macro or the last de-edited macro in an ESERV run.

4. COL Defines location and length of sequence number field.

5. VER Verifies contents of a specific statement.

6. ADD Adds statement(s) at a specified point.

7. DEL Deletes specified statements.

8. REP Replaces specified statement(s).

9. END Indicates last update statement.

10. RST Indicates that sequence number restart a lower number than a number contained in a specified preceding statement of the macro definition.

These control statements can be used to verify and/or update properly identified statements in an edited macro definition.

Verifying and Updating Macro Definitions

Statements are identified either by the sequence number in the identification field or by their position relative to a previous statement that has a sequence number as follows:

```
seq-no[+rel]
```

seq-no sequence number of a source statement, 1 to 8 decimal digits in length, as specified in the COL statement.

rel positive decimal number, 1 to 4 digits in length, representing the position of the desired statement relative to the sequence numbered statements.

COL STATEMENT: specifies the location of the sequence number in the macro definition. If present, this statement must immediately follow the DSPLY, PUNCH or DSPCH statement to which it applies. The format is:

```
)     COL  startcol,x
```

startcol is a decimal number from 73 to 80 which gives the starting position of the sequence number.

x is a decimal number from 1 to 8 giving the length of the sequence number.

If the COL statement is omitted, the default is:

```
)     COL  73,6
```

VER STATEMENT: causes a specific source statement to be verified against the contents of the statement following the VER statement on SYSIPT. Its format is:

```
)     VER  seq-no[+rel],len
```

len is the length of the field to be verified. It is a decimal number with a value of from 1 to 80. Only the first len characters are compared. If the strings do not match, an error message is given.

ADD STATEMENT: adds statements to a source macro definition. Its format is:

```
)     ADD  seq-no[+rel]
```

The source statements following the ADD statement are added to the macro definition after the indicated statement number.

DEL STATEMENT: deletes one or more statements from a macro source definition. Its format is:

```
)     DEL  seq-no[+rel][,seq-no[+rel]]
```

The statement number indicates the first and last in the series to be deleted. Only the first number is required to delete one statement.

REP STATEMENT: replaces one or more statements in a source macro definition with one or more new statements. Its format is:

```
)     REP  seq-no[+rel][,seq-no[+rel]]
```

The source statement(s) following the REP statement replace the statement or series of statements identified by the REP statement.

RST STATEMENT: indicates to ESERV that the sequence numbers of the macro definition restart at a lower number after the statement indicated by the RST statement. The format is:

```
)     RST  seq-no[+rel]
```

Note: If an ADD, DEL or REP operation is performed on the last statement in a sequence number series, then the first statement in the new series must be referenced in the RST statement. See example 3.

END STATEMENT: indicates the end of the update statements on SYSIPT. Its format is:

```
)     END
```

This statement is required for every updating run.

General Rules for Update Control Statements

1. Right parenthesis in column 1 and at least one blank before and after operation code.

2. Any seq-no[+rel] must specify a source statement after the previous seq-no[+rel], except that:

   a. The two numbers specified in a DEL or REP statement may be equal;
   b. the REP, DEL, ADD, or RST statement may reference the same statement as the immediately preceding VER statement.

3. Any seq-no must be greater than or equal to either the last seq-no in the previous statement or the first seq-no in the same statement.

Exception: The first seq-no in the control statement following the RST statement is independent of the seq-no in the RST statement.

Error Handling During Update

The following errors will cause cancellation of the update run.

1. Invalid operands in the COL statement.
2. The COL statement is not the first update control statement.
3. The macro is completely de-edited without all update control statements having been processed.
4. Invalid operands in the RST statement.

In all other cases, an error message is printed. Updating continues with the next control statement, if possible. De-editing of the macro will always be completed.

Example 1: De-editing Without Updating

```
// JOB SAMPL1
// EXEC ESERV
   PUNCH E.MAC1,MAC2
   DSPLY MAC3
   DSPCH E.MAC4,E.MAC5
/*
/&
```

The above coding shows five macros being de-edited in the same run. The resulting listings and/or decks could be used for future updates. Note that the last macro de-edited (MAC5) could also have been updated in this run. All are taken from the macro sublibrary, since E.macname is the default.

Example 2: De-editing with Updating

```
// JOB SAMPLE
// EXEC ESERV
   GENEND
   DSPCH E.MAC1
)  COL 77,4
)  VER 72+1,5
.PP99
)  ADD 72+1
     AIF (&PCH NE 1400).D4
)  DEL 102,103+3
)  REP 245+4,260
NITA  CLC 0(4,REG6),BLANKS
      BE  RETRY6
LYNDA CLC  4(4,REG6),BLANKS
      BE  RETRY7
)  END
/*
/&
```

This example produces a listing on SYSLST and a deck with an END and /* on SYSPCH of the de-edited and updated macro named MAC1. The resulting SYSPCH file (if on tape or disk) could thus be reassigned as SYSIPT for immediate editing by the DOS/VS assembler with EDECK specified in the option card.

In addition, the resulting updated edited macro deck could then, in another step, be cataloged to the E sublibrary via MAINT.

Example 3: Restarting

Sequence number in the macro to be updated:

```
0001
0003
0004
0005
0002
0004
0006
0007
0008
```

Sample update statements:

```
// EXEC ESERV
)  COL 77,4
)   DEL 5
)  RST 5+1
)  ADD 4
   B   NEXT1
)  END
```

In this example, GENCATALS is assumed. Since statement 5 (the last in a series) is to be deleted, the RST statement must reference one past the statement being deleted.

PSERV, PROCEDURE LIBRARY

To request a service function for the procedure library, use the following EXEC control statement:

```
// EXEC PSERV
```

One or more of the three service functions can be requested within a single run. Any number of procedures within the procedure library can be acted upon in this run.

Display

The statement used to display procedures is DSPLY.

Statement Format:

```
{DSPLY procedurename[,procedurename,...]}
{DSPLY ALL                               }
```

procedurename specifies the name of the procedure to be displayed. If more than one procedure is to be displayed, the names are separated by commas. Procedure names are from one to eight alphameric characters, the first of which must be alphabetic or a $.

ALL specifies that the entire procedure library is to be displayed.

For the display function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette. SYSLOG must not be assigned to a printer.

Punch

The control statement used to punch a procedure is PUNCH.

Statement Format:

```
{PUNCH procedurename[,procedurename,...]}
{PUNCH ALL                               }
```

procedurename represents the name of the procedure to be punched. If more than one procedure is to be punched, the names are separated by commas. The procedure name is from one to eight alphameric characters, the first of which must be alphabetic or a $.

ALL specifies that the entire procedure library is to be punched.

When SYSPCH is assigned to a tape or disk unit, each card image is preceded by a stacker-select character.

For the punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSPCH must be assigned to a card punch, a tape unit, a disk unit, or a 3540 diskette. SYSLST must be assigned to a printer, a tape unit, or a disk unit, and SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the module must follow each PUNCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

The punched-card output is acceptable to every function that uses procedure input. Each procedure punched is preceded by a CATALP statement. The last card punched is the /* statement.

Display and Punch

The control statement used to print and punch procedures is DSPCH.

Statement Format:

```
{DSPCH procedurename[,procedurename,...]}
{DSPCH ALL                               }
```

procedurename represents the name of the procedure to be displayed and punched. If more than one procedure is to be displayed and punched, the names are separated by commas. Module names are from one to eight alphameric characters, the first of which must be alphabetic or a $.

ALL specifies that the entire procedure library is to be printed and punched.

When SYSPCH is assigned to a tape or disk unit, or a 3540 diskette, each card image is preceded by a stacker-select character.

For the display and punch function, SYSIPT must be assigned to a card reader, a tape unit, a disk unit, or a 3540 diskette. SYSLST must must be assigned to a printer, a tape unit, a disk unit, or a 3540 diskette. SYSPCH must be assigned to a card punch, a tape unit, or a disk unit. SYSLOG must not be assigned to a printer.

Whenever the same card read punch is assigned to SYSIPT and also to SYSPCH, enough blank cards for punching the procedure must follow each DSPCH control statement. This prevents erroneously punching the cards of a following job step. Extra cards are automatically bypassed.

The last card punched by DSPCH is a /* statement.

Function	Unit	Element	Control Statements

Catalog	Core Image Library	Phase	// OPTION CATAL
			(Linkage Editor control statements and if in card form, the phase to be cataloged)
			/*
			// EXEC LNKEDT

	Relocatable Library	Module	// EXEC MAINT CATALR modulename[,v,m] (module to be cataloged)
	Source Statement Library	Book	// EXEC MAINT CATALS sublib.bookname[,v.m[,C]] (book to be cataloged)
	Procedure Library	Proced.	// EXEC MAINT CATALP procedurename[,VM=v.m][,EOP=yy] [,DATA=NO / YES] (procedure to be cataloged) /+ (or delimiter as specified in EOP parameter)

Compile/ Assemble and Catalog	Relocatable Library	Module	Using a tape file for SYSPCH/SYSIPT with temporary assignments.
			
			// ASSGN SYSPCH,X'cuu'
			CATALR modulename[,v.m]
			PHASE name,origin[,NOAUTO]
			// EXEC COBOL
			(source deck)
			/*
			// MTC WTM,SYSPCH,2
			// MTC REW,SYSPCH
			// ASSGN SYSIPT,X'cuu'
			// EXEC MAINT
			/&
			
			To compile/assemble more than one program and catalog all of them into the relocatable library, use the same setup except that the compilation/assembly control cards (CATALR through /* inclusive) for each program follow the /* of the preceding program. The MTC statements through /& follow the /* of the last program. This facility is not available for RPG and PL/I compilations or for IBM 2314 or IBM 2319 applications.
			
			Using a DASD file for SYSPCH/SYSIPT. (Must always be a permanent assignment.)
			
			// DLBL Balance of information
			// EXTENT required for SYSPCH file
			ASSGN SYSPCH,X'cuu'
			CATALR modulename[,v.m]
			PHASE name,origin[,NOAUTO]
			// EXEC COBOL
			(source deck)
			/*
			CLOSE SYSPCH,X'00D'
			// DLBL Balance of information
			// EXTENT required for SYSIPT file
			ASSGN SYSIPT,X'cuu'
			// EXEC MAINT
			/&

Figure 21. Maintenance Functions, Example (Part 1 of 3)

Function	Unit	Element	Control Statements
			The 'file-ID' in the DLBL statements must be the same in both sets. To compile/assemble more than one program and catalog all of them into the relocatable library, use the same setup except that the compilation/assembly control cards (CATALR through /* inclusive) for each program follow the /* of the preceding program. CLOSE through /& follow the /* of the last program.
Delete	Core Image Library	Phase	// EXEC MAINT DELETC phase1[,phase2,...]
		Program	// EXEC MAINT DELETC prog1.ALL[,prog2.ALL,...]
	Relocatable Library	Module	// EXEC MAINT DELETR module1[,module2,...]
		Program	// EXEC MAINT DELETR prog1.ALL[,prog2.ALL,...]
		Library	// EXEC MAINT DELETR ALL
	Source Statement Library	Book	// EXEC MAINT DELETS sublib.book1[,sublib.book2,...]
		Sub-library	// EXEC MAINT DELETS sublib.ALL
		Library	// EXEC MAINT DELETS ALL
	Procedure Library	Proced.	// EXEC MAINT DELETP procedurename1[,procedurename2,...]
		Library	// EXEC MAINT DELETP ALL
Rename	Core Image Library	Phase	// EXEC MAINT RENAMC oldname,newname[,oldname,newname,...]
	Relocatable Library	Module	// EXEC MAINT RENAMR oldname,newname[,oldname,newname,...]
	Source Statement Library	Book	// EXEC MAINT RENAMS sublib.oldname,sublib.newname[,sublib.oldname,sublib.newname,...]
	Procedure Library	Proced.	// EXEC MAINT RENAMP oldname,newname[,oldname,newname....]
Update	Source Statement Library	Book	// EXEC MAINT UPDATE sublib.bookname,[s.book1],[v.m],[nn]) ADD,) DEL, or) REP statements as required with source statements to be added) END [v.m[,C]]

Figure 21. Maintenance Functions, Example (Part 2 of 3)

Function	Unit	Element	Control Statements
Condense	Core Image Library	Library	// EXEC MAINT CONDS CL

	Relocatable Library	Library	// JOB jobname
			// EXEC MAINT
			CONDS RL

	Source Statement Library	Library	// EXEC MAINT CONDS SL
	Procedure Library	Library	// EXEC MAINT CONDS PL
	Libraries	All	// EXEC MAINT CONDS CL,RL,SL,PL
Set Parameter for Automatic Condense	Libraries	Any or All	// EXEC MAINT CONDL lib=nnnnn[,lib=nnnnn,...] Notes: Values to be substitued for lib: CL -- Core image library PL -- Procedure Library RL -- Relocatable library SL -- Source statement library Values to be substitued for nnnnn: One to five decimal digits, with a maximum value of 65536.

Reallocation	System	Library	// DLBL IJSYSRS,'DOS SYSTEM RESIDENCE FILE'date,code
			// EXTENT SYSRES,balance of extent information
			// EXEC MAINT
			ALLOC CL=cylin(tracks),RL=cylin(tracks),
			SL=cylin(tracks),PL=cylin(tracks)
			
			Notes: The library identifiers are as follows:
			
			CL -- Core image library
			PL -- Procedure Library
			RL -- Relocatable library
			SL -- Source statement library
			
			Values to be substitued for cylin and tracks: any integer.

Note: // JOB, /*, and /& must be included where needed.

Figure 21. Maintenance Function, Example (Part 3 of 3)

Display Unit	Element	Control Statements
Core Image Library	Phase	// EXEC CSERV DSPLY phase1[,phase2,...]
	Program	// EXEC CSERV DSPLY prog1,ALL[,prog2.ALL,...]
	Library	// EXEC CSERV DSPLY ALL
	Directory	// EXEC DSERV DSPLY CD or DSPLYS CD
	Phases(s) with Version and Modification Level	In the standard position: // EXEC DSERV DSPLY[S] CD(phasename) or CD(phasename)
	Phase(s) with Version and Modification Level	In the nonstandard position or higher than DSERV in use: // EXEC DSERV DSPLY[S] CD(phasename,nn) or CD(phasename,nn)
Relocatable Library	Module	// EXEC RSERV DSPLY module1[,module2,...]
	Program	// EXEC RSERV DSPLY prog1.ALL[.prog2.ALL,...]
	Library	// EXEC RSERV DSPLY ALL
	Directory	// EXEC DSERV DSPLY RD or DSPLYS RD
Source Statement Library	Book	// EXEC SSERV DSPLY sublib.book1[.sublib.book2,...]
	Sublibrary	// EXEC SSERV DSPLY sublib1.ALL[,sublib2.ALL,...]
	Library	// EXEC SSERV DSPLY ALL
	Directory	// EXEC DSERV DSPLY SD or DSPLYS SD
Procedure Library	Procedure	// EXEC PSERV DSPLY procedurename1[,procedurename2,...]
	Library	// EXEC PSERV DSPLY ALL
	Directory	// EXEC DSERV DSPLY PD or DSPLYS PD

Figure 22. Service Functions, Example (Part 1 of 3)

Display Unit	Element	Control Statements
Transient Directory	Directory	// EXEC DSERV DSPLY TD or DSPLYS TD
System Directory	Directory	// EXEC DSERV
Directories	All	// EXEC DSERV DSPLY ALL or DSPLYS ALL
Punch Unit	Element	Control Statements
Core Image Library	Phase	// EXEC CSERV PUNCH phase1[,phase2,...]
	Program	// EXEC CSERV PUNCH prog1.ALL[,prog2.ALL,...]
	Library	// EXEC CSERV PUNCH ALL
Relocatable Library	Module	// EXEC RSERV PUNCH module1[,module2,...]
	Program	// EXEC RSERV PUNCH prog1.ALL[.prog2.ALL,...]
	Library	// EXEC RSERV PUNCH ALL
Source Statement Library	Book	// EXEC SSERV PUNCH sublib.book1[,sublib.book2,...][,CMPRSD]
	Sublibrary	// EXEC SSERV PUNCH sublib1.ALL[,sublib2.ALL,...][,CMPRSD]
	Library	// EXEC SSERV PUNCH ALL[,CMPRSD]
Procedure Library	Procedure	// EXEC PSERV PUNCH procedurename1[,procedurename2,...]
	Library	// EXEC PSERV PUNCH ALL

Figure 22. Service Functions, Example (Part 2 of 3)

Display and Punch Unit	Element	Control Statements
Core Image Library	Phase	// EXEC CSERV DSPCH phase1[,phase2,...]
	Program	// EXEC CSERV DSPCH prog1.ALL[,prog2.ALL,...]
	Library	// EXEC CSERV DSPCH ALL
Relocatable Library	Module	// EXEC RSERV DSPCH module1[,module2,...]
	Program	// EXEC RSERV DSPCH prog1.ALL[,prog2.ALL,...]
	Library	// EXEC RSERV DSPCH ALL
Source Statement Library	Book	// EXEC SSERV DSPCH sublib.book1[,sublib.book2,...][,CMPRSD]
	Sublibrary	// EXEC SSERV DSPCH sublib1.ALL[,sublib2.ALL,...][,CMPRSD]
	Library	// EXEC SSERV DSPCH ALL[,CMPRSD]
Procedure Library	Procedure	// EXEC PSERV DSPCH procedurename1[,procedurename2,...]
	Library	// EXEC PSERV DSPCH ALL

Note: // JOB, /*, and /& must be included where needed.

Figure 22. Service Functions, Example (Part 3 of 3)

Copy Unit	Element	Control Statements

Core Image Library	Phase	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYC phase1[,phase2,...]

	Program	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYC prog1.ALL[,prog2.ALL,...]

	Library	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYC ALL

Relocatable Library	Module	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYR module1[,module2,...]

	Program	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYR prog1.ALL[,prog2.ALL,...]

	Library	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYR ALL

Figure 23. Copy Function, Example (Part 1 of 5)

Copy Unit	Element	Control Statements

Source Statement
 	
Library
	Book	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date;code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYS sublib.book1[,sublib.book2,...]

	Sublibrary	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYS sublib1.ALL[,sublib2.ALL,...]

	Library	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYS ALL

Procedure Library	Procedure	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYP procedurename1[,procedurename2,...]

	Library	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPYP ALL

Libraries	All	// ASSGN SYS002,X'cuu'
		// DLBL IJSYSRS,'DOS/VS SYSTEM RESIDENCE
		FILE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		ALLOC CL=cylin(tracks),RL=cylin(tracks),
		SL=cylin(tracks),PL=cylin(tracks)
		COPY ALL

Figure 23. COPY Function, Example (Part 2 of 5)

Copy Unit	Element	Control Statements

Definition of a Private Library (Note 2)	Core Image	// ASSGN SYS003,X'cuu'
		// DLBL IJSYSPC,'user identification of private library',date,code
		// EXTENT SYS003,balance of extent information
		// EXEC CORGZ
		NEWVOL CI=cylin(tracks)

	Relocatable	// ASSGN SYSRLB,X'cuu'
		// DLBL IJSYSRL,'user identification of private library',date,code
		// EXTENT SYSRLB,balance of extent information
		// EXEC CORGZ
		NEWVOL RL=cylin(tracks)

	Source Statement	// ASSGN SYSSLB,X'cuu'
		// DLBL IJSYSSL,'user identification of private library',date,code
		// EXTENT SYSSLB,balance of extent information
		// EXEC CORGZ
		NEWVOL SL=cylin(tracks)

Definition and Creation of a Private Library (Note 2)	Core Image	// ASSGN SYS003,X'cuu'
		// DLBL IJSYSPC,'user identification of private library',date,code
		// EXTENT SYS003,balance of extent information
		// EXEC CORGZ
		NEWVOL CL=cylin(tracks)
		COPYC operands

	Relocatable	// ASSGN SYSRLB,X'cuu'
		// DLBL IJSYSRL,'user identification of private library',date,code
		// EXTENT SYSRLB,balance of extent information
		// EXEC CORGZ
		NEWVOL RL=cylin(tracks)
		COPYR operands

	Source Statement	// ASSGN SYSSLB,X'cuu'
		// DLBL IJSYSSL,'user identification of private library',date,code
		// EXTENT SYSSLB, balance of extent information
		// EXEC CORGZ
		NEWVOL SL=cylin(tracks)
		COPYS operands

Merge System Residence to New System Residence		// ASSGN (statements as required)
		// DLBL IJSYSRS,'NEW SYSTEM RESIDENCE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		MERGE RES,NRS
		COPY statements (COPYC, COPYR, COPYS, COPYP, COPYI) as required

Merge New System Residence to System Residence		// ASSGN (statements as required)
		// DLBL IJSYSRS,'NEW SYSTEM RESIDENCE',date,code
		// EXTENT SYS002,balance of extent information
		// EXEC CORGZ
		MERGE NRS,RES
		COPY statements (COPYI, COPYC, COPYR, COPYS, COPYP) as required

Figure 23. Copy Function, Example (Part 3 of 5)

Copy Unit	Element	Control Statement

Merge System Residence to Private Libraries		// ASSGN (statements as required)
		// DLBL IJSYSRL,'PRIVATE RELOCATABLE
		LIBRARY',date,code
		// EXTENT SYSRLB,balance of extent information
		// DLBL IJSYSSL,'PRIVATE SOURCE STATEMENT
		LIBRARY',date,code
		// EXTENT SYSSLB,balance of extent information
		// DLBL IJSYSCL,'PRIVATE CORE IMAGE
		LIBRARY',date,code
		// EXTENT SYSCLB,balance of extent information
		ASSGN SYSCLB,X'cuu'
		// EXEC CORGZ
		MERGE RES,PRV
		COPY statements (COPYI, COPYR, COPYS, COPYC) as required

Merge New System Residence to Private Libraries		// ASSGN (statements as required)
		// DLBL IJSYSRS,'NEW SYSTEM RESIDENCE',date,code
		// EXTENT SYS002,balance of extent information
		// DLBL IJSYSRL,'PRIVATE RELOCATABLE
		LIBRARY',date,code
		// EXTENT SYSRLB,balance of information
		// DLBL IJSYSSL,'PRIVATE SOURCE STATEMENT
		LIBRARY',date,code
		// EXTENT SYSSLB,balance of extent information
		// DLBL IJSYSCL,'PRIVATE CORE IMAGE
		LIBRARY',date,code
		// EXTENT SYSCLB,balance of extent information
		ASSGN SYSCLB,X'cuu'
		// EXEC CORGZ
		MERGE NRS,PRV
		COPY statements (COPYR, COPYS, COPYC) as required

Merge Private Libraries to System Residence		// ASSGN (statements as required)
		// DLBL IJSYSPR,'PRIVATE RELOCATABLE
		LIBRARY',date,code
		// EXTENT SYS001,balance of extent information
		// DLBL IJSYSPS,'PRIVATE SOURCE STATEMENT
		LIBRARY',date,code
		// EXTENT SYS000,balance of extent information
		// DLBL IJSYSPC,'PRIVATE CORE IMAGE
		LIBRARY',date,code
		// EXTENT SYS003,balance of extent information
		// EXEC CORGZ
		MERGE PRV,RES
		COPY statements (COPYR, COPYS, COPYC) as required

Figure 23. Copy Function, Example (Part 4 of 5)

Copy Unit	Element	Control Statement

Merge Private Libraries to New System Residence		// ASSGN (statements as required)
		// DLBL IJSYSRS,'NEW SYSTEM RESIDENCE',date,code
		// EXTENT SYS002,balance of extent information
		// DLBL IJSYSPR,'PRIVATE RELOCATABLE
		LIBRARY',date,code
		// EXTENT SYS001,balance of extent information
		// DLBL IJSYSPS,'PRIVATE SOURCE STATEMENT
		LIBRARY',date,code
		// EXTENT SYS000,balance of extent information
		// DLBL IJSYSPC,'PRIVATE CORE IMAGE
		LIBRARY',date,code
		// EXTENT SYS003,balance of extent information
		// EXEC CORGZ
		MERGE PRV,NRS
		COPY statements (COPYR, COPYS, COPYC) as required

Merge Private Libraries to Private Libraries		// ASSGN (statements as required)
		// DLBL IJSYSRL,'NEW PRIVATE RELOCATABLE
		LIBRARY',date,code
		// EXTENT SYSRLB,balance of extent information
		// DLBL IJSYSPR,'EXISTING PRIVATE RELOCATABLE
		LIBRARY',date,code
		// EXTENT SYS001,balance of extent information
		// DLBL IJSYSSL,'NEW PRIVATE SOURCE STATEMENT
		LIBRARY',date,code
		// EXTENT SYSSLB,balance of extent information
		// DLBL IJSYSPS,'EXISTING PRIVATE SOURCE STATEMENT
		LIBRARY',date,code
		// EXTENT SYS000,balance of extent information
		// DLBL IJSYSCL,'NEW PRIVATE CORE IMAGE
		LIBRARY',date,code
		// EXTENT SYSCLB,balance of extent information
		ASSGN SYSCLB,X'cuu'
		// DLBL IJSYSPC,'EXISTING PRIVATE CORE IMAGE
		LIBRARY',date,code
		// EXTENT SYS003,balance of extent information
		// EXEC CORGZ
		MERGE PRV,PRV
		COPY statements (COPYR, COPYS, COPYC) as required
		
		To define the private library in the same job step, precede MERGE with a NEWVOL statement, except for COPYC statements.

Notes:

1. // JOB, /*, and /& must be included where needed.

2. The private library can be updated with either a MAINT or a copy MERGE function.

Figure 23. Copy Function, Example (Part 5 of 5)

System Buffer Load (SYSBUFLD)

SYSBUFLD is a special service program that loads the UCB (Universal Character Set Buffer) and the FCB (Forms Control Buffer) for the 3203, 3211, and 5203 printers with buffer load phases. For a 5203 without the universal character set (UCS) feature, only the FCB is loaded. To load the UCB of the printers, the corresponding buffer load phases must reside in the core image library with valid phase names. SYSBUFLD is self-relocating and requires 2K of storage for its operation.

SYSBUFLD is executed in your job stream when it is necessary to change the contents of either the UCB or the FCB. This program is distinct from job control and is initiated by the statement:

```
// EXEC SYSBUFLD
```

SYSBUFLD then reads a control card from SYSIPT, which identifies the unit and the type of buffer load to be performed. The format of this card for the FCB is:

```
FCB SYSxxx[,phasename]
```

The format of the card for the UCB is:

```
UCB SYSxxx[,phasename]
    [,FOLD][,NOCHK]
    [,NULMSG]
```

Note: For each of these formats, a name in the name field must not be used.

Operation

The operation defines the type of buffer load to be performed.

FCB operation loads either a phase in a core image library or an FCB card load (on SYSIPT following the SYSBUFLD control card) into the FCB in the control unit. The EBCDIC characters of the FCB load correspond to the lines of printing for any single form. The maximum number of lines per form is 192 for 3203, 180 for 3211, and 112 for 5203. When a form-skip command is issued to the printer, forms movement is initiated. A character of the form-skip command is compared to the character in the FCB. When a match occurs, forms movement is terminated. Thus, the FCB can adapt the printer to many variable forms.

UCB operation loads the phase from a core image library into the UCB in the control unit. The UCB load corresponds to the print positions on the printer trains. A character sent to the printer for printing is matched against the character in the UCB. When a match occurs, the corresponding train character is printed in the printline position that the output character occupied. Thus, with the use of SYSBUFLD and the trains available, the printers can be adapted to many printing applications.

The following standard UCB load phases are supplied in the core image library:

Printer	Phase Name	Train Type
3203	$$BUCB3	AN
3211	$$BUCB	A11
5203	$$BUCB5	AN

Additional UCB loads (including a copy of the standard one) are supplied in the relocatable library:

Printer	Module Name	Train Type
3203	IJBTRAN	AN or HN
	IJBTRGN	GN
	IJBTRHN	HN
	IJBTRPAN	PCS-AN
	IJBTRPHN	PCS-HN
	IJBTRPN	PN
	IJBTRQNC	QNC
	IJBTRQN	QN
	IJBTRRN	RN
	IJBTRSN	SN
	IJBTRTN	TN
	IJBTRYN	YN
	IJBTRALA	ALA
3211	IJBTRA11	A11
	IJBTRG11	G11
	IJBTRH11	H11
	IJBTRP11	P11
	IJBTRT11	T11
5203	IJBTRAN	AN or HN
	IJBTRGN	GN
	IJBTRPN	PN
	IJBTRHN	HN

The additional UCB loads must be cataloged to a core image library before their execution. They can be assigned any valid phasename.

For dualing feature support during IPL (applies to 3203, 3211, and 5203 printers) develop your own buffer image and catalog it into the core image library with the phasename $$BUCB3 for the 3203, $$BUCB for the 3211, and $$BUCB5 for the 5203. (Refer to the publication _IBM 3211/3216/3811 Component Description and Operating Procedures_, GA24-3543, for further information on the dualing capability).

It is your responsibility to:

- Assemble, link-edit, and catalog any FCB load phases into the core image library and
- Link-edit and catalog any IBM-supplied UCB load phases or assemble, link-edit, and catalog any user-written UCB load phases into the core image library, and
- Mount the new train before the UCB is loaded.

Operands

SYSxxx

The name of the logical unit assigned to a 3203, 3211, or 5203 printer to be loaded is SYSxxx. It must be SYSLST, SYSLOG, or a programmer logical unit.

Note: If SYSLOG is specified in the SYSBUFLD control statement, it must be assigned to a 3203, 3211, or 5203.

phasename

The core image name of the phase containing the applicable buffer load. If FCB is specified and phasename is omitted, an FCB card load from SYSIPT is assumed.

FOLD

FOLD signifies that the UCSB buffer is to be loaded with the folding operation code to permit printing of uppercase for lowercase bit configurations. FOLD is optional and only valid for UCB.

NOCHK

NOCHK prevents the data checks that are generated by the printer because of printline character mismatches with the UCSB. NOCHK is optional and only valid for UCB.

NULMSG

NULMSG signifies that the 80-character verification message is not to be printed on SYSxxx after the buffer is loaded. If NULMSG is not specified after the FCB or UCB has been loaded, the program skips to channel 1, issues a print of the last 80 characters in the phase, and again skips to channel 1. This is repeated for each message.

This message could identify the phase that was loaded. During a UCB load, the train of the printer could be identified by printing a unique character of the train in the message. This would ensure that the mounted train of the printer is compatible with the contents of the UCB.

How to Use SYSBUFLD

The UCB phase format for the 3211 is:

432-character UCB load	80-character field (see _User Written UCB Load Phase_)	80-character verification message

The UCB phase format for the 3203 and 5203 is:

240-character UCB load	80-character verification message

The FCB phase format for the 3203, 3211, and 5203 is:

nnn-character FCB load	80-character verification message

where nnn = 192 for 3203
180 for 3211
112 for 5203

If the FCB phase for a 3211 includes the indexing control byte, the phase format is:

181-character FCB image	80-character verification message

Note 1: Other than $$BFCB (3211), $$BFCB3 (3203), and $$BFCB5 (5203), which are loaded by IPL, no additional FCB loads are supplied by IBM.

Note 2: Both a UCB phase and an FCB phase must include the 80-character verification message.

The FCB card format is:

FCB load
(maximum of 112, 180, or 192 bytes)

```
       Cols 1 8  11       20       29    35        45                  65            80
       ↓      ↓  ↓        ↓        ↓     ↓         ↓                   ↓             ↓

Card 1           1                 2     5         3                   4
Card 2                                   5
Card 3        C           X
       * EXAMPLE OF FCB CARD LOAD 6 LPI, 180 LINES PER PAGE

Card 1 *         1                 2               3                   4
Card 2                                   5
Card 3        C           X
       * EXAMPLE OF FCB CARD LCAD 8 LPI, 180 LINES PER PAGE. (NOTE: * IN
       * COLUMN 1 OF CARD 1 )

Card 1 *C     1           X
       * EXAMPLE OF FCB CARD LCAD FOR PRINTING 8 LPI. CHANNEL 1 IS THE FIRST
       * LINE AFTER THE PAGE PERFORATION. IT IS NECESSARY TO KEEP THE FCB
       * SYNCHRONIZED WITH THE FORM; HOWEVER, IT IS NOT NECCESSARY TO
       * INITIALLY ALIGN THE FIRST LINE OF THE PAGE WITH THE FIRST POSITION
       * OF THE FCB. THIS EXAMPLE IS FOR A FORM WITH 20 LINES PER PAGE WITH
       * PRINTING ON 14 LINES.
```

Figure 24. FCB Card Load Examples for 3211 Printer

The verification message is not allowed. Figure 24 contains examples of FCB card loads for the 3211.

The FCB characters for phase and card format are:

Channel Indication	Phase Format (in Hex)	SYSIPT Card Code
Null (Space)	00	blank
Channel 1	01	1
Channel 2	02	2
Channel 3	03	3
Channel 4	04	4
Channel 5	05	5
Channel 6	06	6
Channel 7	07	7
Channel 8	08	8
Channel 9	09	9
Channel 10	0A	A
Channel 11	0B	B
Channel 12	0C	C
End of FCB	1n	X
8 Lines per inch[1]	1n	* (in col. 1)

[1] (Applies to 3211 only. The 3203 and 5203 have a hardware switch to indicate the line density.)

Note: Channel and End-of-Forms cannot be indicated in the same line for the 3203 or the 5203.

When the FCB phase format is used, bit position 3 (the flag bit) of any channel indication specifies the end of the buffer (for example, X'1C') and, if used for a 3211 in the first buffer position, printing is forced to 8 lines per inch (for example, X'11'). When the FCB is being loaded from SYSIPT, an X must be used to indicate the end of the buffer.

Figure 25 is an example of how to use SYSBUFLD. In the example, the first job step of each job loads the FCB from a core-image phase and the UCB.

```
// JOB ONE
// EXEC SYSBUFLD      (load the FCB and
 FCB SYSLST,PHASE1   UCSB for PROG001)
 UCB SYSLST,UCBPH001,FOLD,NOCHK,NULMSG
/*
// EXEC PROG001
/*
/&
// JOB TWO
// EXEC SYSBUFLD         (load the FCB and
 UCB SYSLST,UCBPH002  UCSB for PROG002)
 FCB SYSLST,PHASE2
/*
// EXEC PROG002
/*
/&
```

Figure 25. Example of System Buffer Load

USER-WRITTEN FCB LOAD PHASE

Figure 26 shows how PHASE001 might be defined, assembled, and cataloged to the core image library for an FCB load.

```
// JOB FCBXMPLE
// OPTION CATAL,DECK
// EXEC ASSEMBLY
 PUNCH ' PHASE   PHASE001,+0'
 START
 DC X'01' CHANNEL 1 & 1ST LINE OF PAGE *
 DC XL64'00' 64 LINES OF PAGE
 DC X'1C' CHANNEL 12 & LAST LINE PAGE
 DC XL114'00' UNUSED BUFFER POSITIONS
 DC CL80'FCB PHASE001 LOADED' MESSAGE
 END
/*
// EXEC LNKEDT
/&
```

*Channel and End-of-Forms cannot be indicated in the same line for the 3203 or 5203.

Figure 26. FCB Load Program

An example of the instructions that should be written for these printers is given below:

```
  DC XL63'00'  63 LINES OF PAGE
  DC X'0C'     CHANNEL 12
  DC X'10'     END OF FORMS
```

USER-WRITTEN UCB LOAD PHASE

If your 3203, 3211, or 5203 uses a special train configuration, then you must assemble, link-edit, and catalog the UCB load phase for the special train into the core image library. The UCB load phase format for the 3211 is:

Position	Comment
1-432	Train image. The hexadecimal equivalent of all characters on the train.
433-447	Zeros.
448-511	Associative field. This is used by the 3811 Control Unit to check for invalid hexadecimal characters sent to the 3211. For a complete description of this field and how to prepare it, see IBM 3211/3216/3811 Component Description and Operating Procedures, GA24-3543.
512	Zero.
513-592	Verification message.

The UCB load phase format for the 3203 and 5203 is:

Position	Comment
1-240	Train image. The hexadecimal equivalent of all characters on the train.
241-320	Verification message.

SYSBUFLD MESSAGES

Appropriate messages are issued when an invalid parameter specification is made or when a required parameter is omitted. With the exception of an invalid phasename, all errors on the control card may be corrected through SYSLOG. An invalid phasename cancels the job.

SYSBUFLD UNDER POWER/VS

POWER/VS supports spooling of 3203, 3211, and 5203 UCB and FCB load commands as given by the service program SYSBUFLD.

When jobs are executed under POWER/VS, pages are formatted by posting channel 9 and channel 12 as specified in the LINETAB parameter of the * $$ PRT statement. (For a description of the * $$ PRT statement refer to the chapter POWER/VS.) When the output of the job is finally printed, skip operations are performed according to the current contents of the forms control buffer. The contents of the buffer must therefore correspond to the specifications of the LTAB operand.

LOADING AN FCB

The contents of an FCB may be changed anywhere within a job entry. The job to load an FCB may also be submitted as a single job entry.

A job entry that loads a special buffer should reset the buffer to its normal contents before the job entry is terminated. If that is not possible, the FCB may be reset by submitting a separate FCB load job entry that resets the buffer to normal contents.

LOADING A UCB

A UCB may be loaded under control of POWER/VS by submitting the following statements anywhere within a job entry:

```
[// JOB]
 // EXEC SYSBUFLD
    UCB SYSLST,phasename,FOLD,NOCHK
[/&]
```

This job may also be submitted as a separate job entry.

A UCB load is executed as soon as it is encountered in the print-line image stream. It does not force POWER/VS to stop printing. Note: NOCHK must always be specified in the UCB card to avoid data checks in the print writer routine.

Appendix A: Job Control Summary

Appendix A contains a summary of the following:

- Job Control Statements (JCS) which must be preceded by // blank in positions 1, 2, and 3.
- Job Control Commands (JCC).
- Attention Routine Commands (AR).

The POWER/VS statements and commands are summarized in Appendix C.

Name	Operation	Operand	Remarks
	ALLOC Valid for JCC, AR	F1=nK[,F2=nK][,F3=nK][,F4=nK]	Allocates foreground program areas in the virtual address area. Value of n is an even number F and either 0 (inactive) or a minimum of 64 (active). The order of operands is arbitrary. At least one operand must be specified.
	ALLOCR Valid for JCC, AR	BGR=nK [,F1R=nK] [,F2R=nK] [,F3R=nK] [,F4R=nK]	Allocates real address area among foreground and back-ground programs. Value of n is zero or even number. The order of operands is arbitrary. At least one operand must be specified.
	ALTER Valid for AR	xxxxxx	Alters 1 to 16 bytes of virtual storage. xxxxxx is the hexadecimal address where alteration is to start.
[//]	ASSGN Valid for JCS, JCC	For any device SYSxxx,{X'cuu' [,TEMP] (address-list) [,PERM] SYSyyy UA IGN} For disks SYSxxx,{X'cuu' [,TEMP][,VOL=no][,SHR] (address-list) [,PERM] SYSyyy DISK 2311 2314 3330 3340} For diskettes SYSxxx,{X'cuu' \ [,TEMP] (address-list) [,PERM] SYSyyy DISKETTE 3540} For tapes SYSxxx,{X'cuu' [,X'ss'][,TEMP][VOL=no] (address-list) [,ALT][,PERM] SYSyyy TAPE 2400T7 2400T9 3410T7 3410T9 3420T7 3420T9}	SYSxxx can be SYSRDR SYSIPT SYSIN SYSPCH SYSLST SYSOUT SYSLOG SYSLNK SYSREC SYSRLB SYSSLB SYSCLB SYSnnn X'cuu': c = 0-6 uu = 00-FE (0-254) in hex address-list: a list of up to seven device addresses in the form (X'cuu',...,X'cuu') UA: unassign IGN: unassign and ignore (invalid for SYSCLB, SYSRDR, SYSIPT, and SYSIN) SYSyyy: any system or programmer logical unit device-class: READER, PRINTER, PUNCH, TAPE, DISK or DISKETTE device-type: device code of any supported device X'ss': used for magnetic tape only.

ss	Bytes per inch	Parity	Translate Feature	Convert Feature
10	200	odd	off	on
20	200	even	off	off
28	200	even	on	off
30	200	odd	off	off
38	200	odd	on	off
50	556	odd	off	on
60	556	even	off	off
68	556	even	on	off
70	556	odd	off	off
78	556	odd	on	off
90	800	odd	off	on
A0	800	even	off	off
A8	800	even	on	off
B0	800	odd	off	off
B8	800	odd	on	off

Figure 27. Job Control Summary (Part 1 of 8)

Name	Operation	Operand	Remarks
	(ASSGN continued)	For printers SYSxxx, X'cuu' | (address-list) | SYSyyy | PRINTER | 1403 | 1403U | 1443 | 3203 | 3211 | 5203 | 5203U [,TEMP | ,PERM] For card (read) punches SYSxxx, X'cuu' | (address-list) | SYSyyy | PUNCH | 1442N1 | 1442N2 | 2520B1 | 2520B2 | 2520B3 | 2540P | 2560 [,H1 | ,H2] | 2596 | 3525P | 3525RP | 5425 [,H1 | ,H2] [,TEMP | ,PERM] For card readers SYSxxx, X'cuu' | (address-list) | SYSyyy | READER | 1442N1 | 2501 | 2520B1 | 2540R | 2560 [,H1 | ,H2] | 2596 | 3504 | 3505 | 3525RP | 5425 [,H1 | ,H2] [,TEMP | ,PERM]	C0 800 single density 9 track tape C0 1600 single density 9 track tape C0 1600 dual density 9 track tape C8 800 dual density 9 track tape D0 6250 single density 9 track tape D0 6250 dual density 9 track tape ALT: specifies alternate tape unit. H1: specifies input hopper one for input on 2560 or 5425. Is assumed if neither H1 nor H2 is specified. H2: specifies input hopper two for input on 2560 or 5425 (invalid for programmer units and POWER). PERM: the assignment is permanent. TEMP: the assignment is temporary. VOL=no: volume serial number of the tape or disk required. SHR: indicates the share option for disk devices.
	BATCH Valid for AR	BG | F1 | F2 | F3 | F4	Start, or continue processing. If specified partition was halted by STOP command, it is made active; same function as START command.
	CANCEL Valid for AR	BG | F1 | F2 | F3 | F4	Cancels execution of current job in specified partition.
	CANCEL Valid for JCC	blank	Cancels execution of current job.

Figure 27. Job Control Summary (Part 2 of 8)

Name	Operation	Operand	Remarks
[//]	CLOSE Valid for JCS, JCC	SYSxxx [,X'cuu' [,X'ss'] ,UA ,IGN ,ALT ,SYSyyy]	SYSxxx } for magnetic tape — for DASD (JCC only) SYSyyy } SYSPCH — SYSIN SYSLST — SYSRDR SYSOUT — SYSIPT SYSnnn — SYSPCH X'cuu',X'ss',UA,IGN,ALT: Values as described in ASSGN command.
//	DATE Valid for JCS	mm/dd/yy or dd/mm/yy	mm: month (01-12) dd: day (01-31) yy: year (00-99)
//	DLAB Valid for JCS	'label fields 1-3', C xxxx,yyddd,yyddd,'system code'[,type]	'label fields 1-3': first three fields of Format 1 DASD file label. Is a 51-byte character string, contained within apostrophes and followed by a comma. Entire 51-byte field must be contained in the first of the two statements. Field 1 is the file name (44-byte alphameric); field 2 is the format identifier (1-byte numeric); field 3 is the file serial number (6-byte alphameric). C: any nonblank character in column 72. xxxx: volume sequence number (4-digit numeric). Must begin in column 16 of the continuation statement. Columns 1-15 are blank. yyddd,yyddd: file creation date followed by file expiration date. Each is 5-digit numeric. 'system code': not required. When used, a 13-character string, within apostrophes. type: SD, DA, ISC, or ISE. If omitted, SD is assumed.
//	DLBL Valid for JCS	filename, ['file-ID'] , [date], [codes] , [DSF], [BUFSP=n], [CAT=filename] (See Note 1)	filename: one to seven alphameric characters, the first of which must be alphabetic. 'file-ID': one to forty-four alphameric characters (one to eight alphameric characters for the 3540 diskette). date: one to six characters (yy/ddd for expiration date or yyddd for retention period). codes: two to four alphabetic characters. DSF: data secured file. Invalid for 3540 diskette and for VSAM. BUF SP=n: for VSAM files. One to six digits (0—999999). CAT=filename for VSAM files. For filename specify one to seven alphameric characters.
	DSPLY Valid for AR	XXXXXX	Displays 16 bytes of virtual storage.
	DUMP Valid for AR	{ Blank S BG Fn BGS FnS PDAREA address,address } [{ BG Fn }] *	Dumps specified areas of virtual storage. *Parameter causes dump on the SYSLST assigned to the specified partition. Default is BG SYSLST. blank: general registers plus all real and virtual partitions currently occupied by programs. S: general registers, all real and virtual partitions currently occupied by programs, and supervisor area. BG,Fn: applicable real or virtual partition currently occupied by program, and associated registers. n = 1, 2, 3, or 4. BGS,FnS: applicable real or virtual partition currently occupied, registers, and supervisor area. n = 1, 2, 3, or 4. PDAREA: PD table,PD area, and AAA. address, address: start and end addresses of storage are to be dumped. Associated general registers are also dumped.

Not81: If the DLBL and EXTENT statements for a private core image library are in the input stream (that is, the information isnot contained on the label cylinder), they must precede the ASSGN SYSCLB command.

Figure 27. Job Control Summary (Part 3 of 8)

Name	Operation	Operand	Remarks
	DVCDN Valid for JCC	X'cuu'	X'cuu': c = 0-6 in hex uu = 00-FE (0-254) in hex
	DVCUP Valid for JCC	X'cuu'	X'cuu': c = 0-6 in hex uu = 00-FE (0-254) in hex

	END or ENTER
	Valid for
	JCC, AR
		blank	end of SYSLOG communications
			END key for the 3210 and 3215 printer keyboards
			ENTER key for DOC

	ENDSD Valid for AR	blank	Terminates execution of SD aids program.
[//]	EXEC Valid for JCS and JCC	[[PGM=]progname][,REAL][,SIZE=size] PROC=procname[,OV]	PGM=progname: one to eight alphameric characters. Used only if the program is in the core image library. REAL: the respective program is to be executed in real mode. SIZE=size: can be the following: nK: size of area required AUTO: take program size (AUTO,nK): take program size plus nK. PROC=procname: name of cataloged procedure to be retrieved. One to eight alphameric characters. OV: indicates that overwrite statements follow EXEC statement.,
//	EXTENT Valid for JCS	[symbolic-unit], [serial-number],[type], [sequence-number], [relative-track], [number-of-tracks], [split-cylinder-track], [B=bins] (See Note 1)	symbolic unit: six alphameric chafacters serial number: one to six alphameric characters type: one numeric character sequence number: one to three numeric characters relative track: one to five numeric characters number of tracks: one to five numeric characters split cylinder track: one or two numeric characters bins: one or two numeric characters

	HOLD
	Valid for
	JCC
		{ F1[,F2][,F3][,F4] }
		{ F4[,F3][,F2][,F1] }
			Used to deactivate a partition.

	IGNORE Valid for AR and JCC	blank	Ignore abnormal condition.
//	JOB Valid for JCS	jobname [accounting information]	jobname: one to eight alphameric characters accounting information: one to 16 characters

//	LBLTYP
	Valid for
	JCS
		{TAPE[(nn)]}
		{NSD (nn)}
			TAPE: Used when tape files reqoiring label information are to be processed and no nonsequential disk files are to be processed.
			(nn): Optional and is present only for future expansion (it is ignored by job control).
			NSD: Nonsequential disk files are to be processed.
			(nn): Largest number of extents per single file.

	LFCB valid for: AR	X'cuu',phasename [,FORMS = xxxx] [,LPI = n] [,NULMSG]	Causes the FCB image of printer X'cuu' to be loaded.
[//]	LISTIO Valid for JCS JCC	SYS PROG BG Fn ALL SYSxxx UNITS DOWN UA X'cuu'	Causes listing of I/O assignments on SYSLST for JCS and SYSLOG for JCC. Fn = Physical units assigned to all logical units of foreground partition. n = 1, 2, 3, or 4.

Figure 27. Job Control Summary (Part 4 of 8)

Name	Operation	Operand	Remarks
	LOG Valid for JCC AR	blank	Causes logging of job control statements on SYSLOG.
	LUCB valid for: AR	X'cuu',phasename [,FOLD] [,NOCHK] [,TRAIN = xxxxxx] [,NULMSG]	Causes the UCB image of printer X'cuu' to be loaded.
	MAP Valid for JCC AR	blank	Produces (on SYSLOG) storage areas allocated to partitions.
	MODE Valid for AR	{ IR CR CE,cuu [,I[,xx,y] / ,D[,xx,y] / ,N] R STATUS HIR / ECC[,M / ,C] *{[,R / ,Q / ,TH]} [,E=eeee][,T=tttt]* }	Changes the mode of operation, changes the EFL threshold values, and gives status information. *Note: When HIR or ECC is specified, at least one of the optional operands must be selected. TH is only valid for the Model 145 when ECC,C is specified with the MODE command.
	MSG Valid for AR	{ BG F1 F2 F3 F4 }	Transfer control to an operator communications routine of a particular partition previously activated by a STXIT command.
[//]	MTC Valid for JCS JCC	opcode, {SYSxxx / X'cuu'} [,nn]	opcode: BSF, BSR, DSE, ERG, FSF, FSR, REW, RUN, or WTM. SYSxxx: any logical unit. X'cuu': (Only valid for JCC) c = 0-6, uu = FE (0-254) in hex nn: decimal number (01-99).
	NEWVOL Valid for AR	[BG / Fn]	Indicates that a new volume has been mounted for the specified partition. Fn can be specified as F1, F2, F3, or F4.
	NOLOG Valid for JCC AR	blank	Suppresses logging of job control statements and commands on SYSLOG, except ALLOC, ALLOCR, DVCDN, DVCUP, EOJ (/&), EOP (/+), HOLD, JOB, MAP, PAUSE,* and STOP. AR affects all partitions in which NOLOG is issued.
//	OPTION Valid for JCS	option 1[,option 2...]	option: can be any of the following LOG Log control statements on SYSLST NOLOG Suppress LOG option DUMP Dump registers, supervisor area, and temporary real or virtual partition on SYSLST in case of abnormal program end. PARTDUMP Dump registers, selected supervisor control blocks, and temporary real or virtual partition on SYSLST in case of abnormal program end. NODUMP Suppress DUMP or PARTDUMP option LINK Write output of language translator on SYSLNK for linkage editing NOLINK Suppress LINK option DECK Output object module on SYSPCH NODECK Suppress DECK option EDECK Punch source macro definitions on SYSPCH NOEDECK Suppress EDECK option ALIGN Align constants and data areas on boundaries NOALIGN Suppress ALIGN option LIST Output listing of source module on SYSLST NOLIST Suppress LIST option LISTX Output of object module on SYSLST SYM Punch symbol deck on SYSPCH NOSYM Suppress SYM option XREF Output symbolic cross-reference list on SYSLST NOXREF Suppress XREF option ERRS Output listing of all errors in source program on SYSLST NOERRS Suppress ERRS option ACANCEL Cancel job if attempt to assign device is unsuccessful

Figure 27. Job Control Summary (Part 5 of 8)

Name	Operation	Operand	Remarks
	(OPTION continued)		NOACANCEL Await operator action if a device cannot be assigned CATAL Catalog program or phase in core image library after completion of Linkage Editor run STDLABEL Causes all DASD or tape labels to be written on the standard label track USRLABEL Causes all DASD or tape labels to be written on the user label track PARSTD Causes all DASD or tape labels to be written on the partition standard label track 48C 48 - character set 60C 60 - character set SYSPARM='string' specifies a value for assembler system variable symbol and SYSPARM SUBLIB=DF Causes assembler and ESERV program to retrieve non-edited macros and copy-books from D-sublibrary and edited macros from the F-sublibrary of the source statement library. SUBLIB=AE Causes assembler and ESERV program to retrieve non-edited macros from the A-sublibrary and edited macros from the E-sublibrary of the source statement library.
[//]	OVEND Valid for JCS, JCC	[comments]	Indicates end of overwrite statements for a cataloged procedure.
[//]	PAUSE Valid for JCS	[comments]	Causes pause immediately after processing this statement. PAUSE statement is always printed on SYSLOG. If no 3210, 3215, or DOC is available, the statement is ignored.
	PAUSE Valid for AR, JCC	{BG F4 F3} [,EOJ] F2 F1}	Causes pause at end of current job step or at end of job.
	PRTY Valid for AR	[P1,...,Pn]	P = BG, F1, F2, F3, F4. Allows the operator to display or change the priority of partitions.
[//]	RESET Valid for JCS JCC	{SYS PROG ALL SYSxxx}	Resets I/O device assignments. SYSxxx: SYSIN or SYSOUT cannot be specified.
	ROD Valid for JCC	blank	Causes all SDR counters for all nonteleprocessing devices on the recorder file on SYSREC to be updated from the SDR counters in storage.
//	RSTRT Valid for JCS	SYSxxx,nnnn[,filename]	SYSxxx: symbolic unit name device on which the checkpoint records are stored. Can be SYSnnn. nnnn: four character identification of the checkpoint record to be used for restarting. filename: symbolic name of the DASD file to be used for restarting.
	SET Valid for JCC	[,UPSI=n1][,LINECT=n2] [,RCLST=n3][,RCPCH=n4] [,RF=n5][,DATE=n6] [,HC=n7][,SVA=n8][,SDL=n9]	Note: The order of the operands is arbitrary. n1 = 1 to 8 digits of 0, 1, or X. (X = ignore; unspecified rightmost positions are assumed to be X.) n2 = standard number of lines (30-99) for output on each page of SYSLST. n3 = decimal number (default = 1000) indicating minimum number of SYSLST disk records remaining to be written before operator warning. n4 = decimal number (100-65535) indicating minimum number of SYSPCH disk records remaining to be written before operator warning. n5 = {YES CREATE} defines status of recorder file (IJSYSRC) on SYSREC. n6 = {mm'dd'yy dd'mm'yy} date defaults to system generation mm: month (01-12) dd: day (01-31) yy: year (00-99)

Figure 27. Job Control Summary (Part 6 of 8)

Name	Operation	Operand	Remarks
	(SET continued)		n7 = {YES NO CREATE} Hard copy file exists. No recording performed. Creates a hard copy file. n7 defines the status of the hard copy file on the Model 115/125 SYSREC. n8 = storage size in the format (nK,nK) for SVA and GETVIS area, respectively. n9 = specify CREATE to have the system directory list (SDL) built in the SVA.
	START Valid for AR	[BG F1 F2 F3 F4]	BG: Job control reads next control statement from SYSLOG. F1-F4: Specifies that an inactive or stopped foreground partition is to be started. (See also BATCH command.)
	STOP Valid for JCC	blank	Indicates that there are no more jobs to be executed in partition specified. Not valid in a single partition system.
//	TLBL Valid for JCS	filename,['file-ID'],[date], [file-serial-number], [volume-sequence-number], [file-sequence-number], [generation-number], [version-number] Note : For ASCII file processing the fourth and fifth operands are called set identifier and file section number, respectively.	filename: one to seven alphameric characters, the first of which must be alphabetic. 'file-ID': one to 17 alphameric characters. date: one to six numeric characters (yy/ddd or dddddd). {[file serial number (EBCDIC): one to six alphameric characters] [set identifier (ASCII): six alphameric characters]} {[volume sequence number (EBCDIC)] [file section number (ASCII)]} one to four numeric characters file sequence number: one to four numeric characters. generation number: one to four numeric characters. version number: one to two numeric characters.
	TPBAL Valid for AR	[n]	n = number of partitions in which processing can be delayed (0,1,2, . . . , number of partitions minus one). Allows the operator to display or alter the status of the Teleprocessing Balancing function.
//	TPLAB Valid for JCS	{'label fields 3-10' 'label fields 3-13' C}	'label fields 3-10': Indicated fields of the standard tape file label for either EBCDIC or ASCII files. A 49-byte character string, contained within apostrophes. 'label fields 3-13': Same as above, except that character string is 69 bytes long, requiring a continuation character in column 72 of first line. C: Any nonblank character in column 72.
	UCS Valid for JCC	SYSxxx,phasename[,FOLD][,BLOCK] [,NULMSG]	Causes the 240-character universal character set contained in the core image library phase specified by phasename to be loaded as buffer storage in the IBM 2821 Control Unit. If SYSxxx is assigned to a 1403 or 5203 printer, the UCS feature must be present.
	UNBATCH Valid for JCC	blank	Terminates foreground processing.
	UPSI Valid for JCS	nnnnnnnn	n: one to eight characters of 0, 1, or X. Unspecified rightmost positions are assumed to be X.
	VOL Valid for JCS	SYSxxx,filename	SYSxxx: symbolic unit name. filename: one to seven alphameric characters, the first of which must be alphabetic

Figure 27. Job Control Summary (Part 7 of 8)

Name	Operation	Operand	Remarks
//	XTENT Valid for JCS	type sequence,lower,upper 'serial no.',SYSxxx[,B_2]	type: 1 for data area (no split cylinder) 2 for overflow area (for indexed sequential file) 4 for index area (for indexed sequential file) 128 for data area (split cylinder) sequence: sequence number of extent within multiextent file. Can be 0 to 255. lower: lower limit of extent in the form $B_1C_1C_1C_2C_2H_1H_2H_2$ where: B_1 = 0 for 2311 or 2314/2319; 0-9 for 2321 C_1C_1 = 00 for 2311 or 2314/2319; 00-19 for 2321 $C_2C_2C_2$ = 000-199 for 2311 or 2314/2319, 000-009 for 2321 H_1 = 0 for 2311 or 2314/2319; 0-4 for 2321 H_2H_2 = 00-09 for 2311; 00-19 for 2321 or 2314/2319. Note that the last four strips of subcell 19 are reserved for alternate tracks for 2321. upper: upper limit of extent in the same form as for lower limit. 'serial no.': 6-alphameric-character volume serial number contained within apostrophes. SYSxxx: symbolic unit name. B_2: 0 for 2311 or 2314/2319; 0-9 for 2321.
//	ZONE Valid for JCS	{EAST WEST} /hh/mm	EAST: A geographical position east of Greenwich. WEST A geographical position west of Greenwich. hh: A decimal value in the range 0-12. mm: A decimal value in the range 0-59.
/+	Valid for JCS	[comments]	Indicates end of procedure.
/*	Valid for JCS	ignored	Indicates end of data.
/&	Valid for JCS	[comments]	Indicates end of job (EOJ).
*	Valid for JCS	comments	Comment card. Column 2 must be blank.

Figure 27. Job Control Summary (Part 8 of 8)

Appendix B: Linkage Editor Summary

FORMAT OF THE ESD CARD

Card
Columns

1 Multiple punch (12-2-9). Identifies this as a loader card.

2 - 4 ESD -- External Symbol Dictionary card.

11 - 12 Number of bytes of information contained in this card.

15 - 16 External symbol identification number (ESID) of the first SD, PC, CM or ER on this card. Relates the SD, PC, CM or ER to a particular control section.

17 - 72 Variable information.

8 positions - Name
1 position - Type code hex '00', '01', '02', '04', '05', or '0A' to indicate SD, LD, ER, PC, CM, or WX, respectively.
3 positions - Assembled origin
1 position - Blank
3 positions - Length, if an SD-type, CM-type, or a PC-type. If an LD-type, this field contains the external symbol identification number (ESID) of the SD containing the label.

73 - 80 May be used by the programmer for identification.

FORMAT OF THE TXT CARD

Card
Columns

1 Multiple punch (12-2-9). Identifies this as a loader card.

2 - 4 TXT -- Text card.

6 - 8 Assembled origin (address of first byte to be loaded from this card).

11 - 12 Number of bytes of text to be loaded.

15 - 16 External symbol identification number (ESID) of the control section (SD or PC) containing the text.

17 - 72 Up to 56 bytes of text -- data or instructions to be loaded.

73 - 80 May be used for program identification.

FORMAT OF THE RLD CARD

Card
Columns

1 Multiple punch (12-2-9). Identifies this as a loader card.

2 - 4 RLD -- Relocation List Dictionary card.

11 - 12 Number of bytes of information contained in this card.

17 - 72 Variable information (multiple items).

a. Two positions - (relocation identifier) pointer to the ESID number of the ESD item on which the relocation factor of the contents of the address constant is dependent.

b. Two positions - (position identifier) pointer to the ESID number of the ESD item on which the position of the address constant is dependent.

c. One position - flag indicating type of constant, as follows:

Bits

0-2 ignored

3 0 - a nonbranch type load constant

1 - a branch type load constant

4-5 00 - load constant length = 1 byte

01 - load constant length = 2 bytes

10 - load constant length = 3 bytes

11 - load constant length = 4 bytes

6 0 - relocation factor is to be added

1 - relocation factor is to be subtracted

7 0 - Next load constant has different R and P identifiers; therefore, both R and P must be present.

1 - Next load constant has the same R and P identifiers; therefore they are both omitted.

Five significant bits of this byte are expanded in the RSERV printout.

d. Three positions - assembled origin of load constant.

73 - 80 May be used for program identification.

FORMAT OF THE END CARD

Card
Columns

1 Multiple punch (12-2-9). Identifies this as a loader card.

2 - 4 END

6 - 8 Assembled origin of the label supplied to the assembler in the END card (optional).

15 - 16 ESID number of the control section to which this END card refers (only if 6-8 present).

17 - 22 Symbolic label supplied to the assembler if this label was not defined within the assembly.

29 - 32 Control section length (if not specified in last SD or PC).

73 - 80 Not used.

FORMAT OF THE REP (USER REPLACE) CARD

Card
Columns

1 Multiple punch (12-2-9). Identifies this as a loader card.

2 - 4 REP -- Replace text card.

5 - 6 Blank.

7 - 12 Assembled address of the first byte to be replaced (hexadecimal). Must be right justified with leading zeros if needed to fill the field.

13 Blank.

14 - 16 External symbol identification number (ESID) of the control section (SD) containing the text (hexadecimal). Must be right justified with leading zeros if needed to fill the field.

17 - 70 From 1 to 11 4-digit hexadecimal fields separated by commas, each replacing two bytes. A blank indicates the end of information in this card.

71 - 72 Blank.

73 - 80 May be used for program identification.

Appendix C: POWER/VS Summary

Figures 28, 29, and 30 summarize the following statements and commands in alphabetical order:

1. POWER/VS Job Entry Control Language (JECL)
2. Central Operator Commands
3. POWER/VS RJE Terminal Commands.

Operation	Operand	Remarks
* $$ CTL	[CLASS={A class}]	Specifies a default input class that is assigned to all jobs whose input class was not assigned in an * $$ JOB statement. class = A-Z (partition-independent), or 0-4 (partition-dependent)
* $$ DATA	name	Specifies the name of the corresponding * $$ DATA statement in the source statement library book where data is to be inserted. name = 1-8 alphameric characters.
* $$ EOJ	(no operand)	Indicates the end of a POWER/VS job.
* $$ JOB	Positional Format [AUTONAME jobname] ,[D disposition] ,[priority] ,[class}	Indicates the beginning of a POWER/VS job and provides handling information. jobname = 2-8 alphameric characters disposition = D: delete after processing H: hold K: keep after processing L: leave in queue priority = 0-9 (9 is the highest priority). class = A-Z (partition-independent) 0-4 (partition-dependent) user-information = 1-16 bytes; may be specified in quotes, so that blanks may be included.
* $$ JOB	Keyword Format [JNM={AUTONAME jobname}] [,DISP={D disposition}] [,PRI=priority] [,CLASS=class] [,USER=user-information]	

Figure 28. Summary of POWER/VS JECL Statements (Part 1 of 3)

Operation	Operand	Remarks
* $${LST PRT}	Positional Format [D [A disposition class]] ,[ƀƀƀƀ forms-number] [1 ,number-of-copies tape-devaddr] ,[norbm1] ,[linetab]	Provides handling information for printed output. disposition = D: delete after processing H: hold K: keep after processing L: leave in queue N: do not intercept R: delete after processing T: spool to tape

* $${LST
PRT}
	Keyword Format
	[DISP={D
	disposition}]
	[,CLASS={A
	class}]
	[,REMOTE=remid]
	[,FNO={ƀƀƀƀ
	forms-number}]
	[,JSEP=sep]
	[,COPY={1
	number-of-copies}]
	[,TADDR=tape-devaddr]
	[,RBM=(norbm1,norbm2)]
	[,LTAB=linetab]
	[,RBS=norbs]
	[,UCS=(phasename[,option]
	[,FCB=phasename]
	[,LST=listaddr]
		class = A-Z
		remid = 0-200
		0 = central operator
		forms-number = 1-4 alphameric characters
		sep = 0-9
		number-of-copies = 0-99
		tape-devaddr = cuu, or X'cuu', or cuu,X'ss',
		or X'cuu',ss, or cuu,ss, or X'cuu',X'ss'
		norbm1 = 1-6 digits
		norbm2 = 1-6 digits
		linetab = 26 digits; specifies the carriage control tape format
		norbs = 1-6 digits
		phasename = 1-8 characters
		option = F, C, FC, or CF)]
		listaddr = SYSxxx (SYSLST or any programmer logical unit) or cuu, or X'cuu'

* $$ PUN	Positional Format [D [A disposition class]] ,[ƀƀƀƀ forms-number] [1 ,number-of-copies tape-devaddr] ,[norbm1]	Provides handling information for punched output. disposition = D: delete after processing H: hold I: return to input K: keep after processing L: leave in queue N: do not intercept R: delete after processing T: spool to tape

* $$ PUN	Keyword Format
	[DISP={D
	disposition}]
	[,CLASS={A
	class}]
	[,REMOTE=remid]
	[,FNO={ƀƀƀƀ
	forms-number}]
	[,JSEP=sep]
	[,COPY={1
	number-of-copies}]
	[,TADDR=tape-devaddr]
	[,RBM=(norbm1,norbm2)]
	[,RBS=norbs]
	[,PUN=punaddr]
		class = A-Z
		forms-number = 1-4 alphameric characters
		remid = 1-200; 0 = central location
		sep = 0-9
		number-of-copies = 0-99
		tape-devaddr = cuu, or X'cuu', or cuu,X'ss', or X'cuu',X'ss', or cuu,ss, or X'cuu',ss
		norbm1 = 1-6 digits
		norbm2 = 1-6 digits
		norbs = 1-6 digits
		punaddr = SYSxxx: SYSPCH or any programmer logical unit (or cuu or X'cuu')

Figure 28. Summary of POWER/VS JECL Statements (Part 2 of 3)

Operation	Operand	Remarks
	Positional Format	
* $$ RDR	[physical-unit-number] ,['file-id'] ,[1 number-of-diskettes]	Inserts a diskette file into the input stream.
	,[S]	S = volume sequence checking
	Keyword Format	
* $$ RDR	[DEV=physical-unit-number]	Physical unit-number = X'cuu' or cuu
	[,FID='file-id']	file-id = 1-8 alphameric characters
	[,NOD={1 number-of-diskettes}]	number-of-diskettes = 1-255
	[,VSC={NO YES}]	volume sequence checking
	[,VER={NO YES}]	check on file verification
* $$ SLI	[sublib.]bookname	Inserts data from a sublibrary into the job stream. sublib = A-Z or 0-9, $, #, or @
* $$/*	(no operand)	Indicates end of job step.
* $$/&	(no operand)	Indicates end of job.

Figure 28. Summary of POWER/VS JECL Statements (Part 3 of 3)

Operation	Operand	Remarks
{PACCOUNT J}	{tapeaddr[,filename] DISK,filename DEL}	Saves accumulated account file records. If no operand, the account file is spooled to the punch queue with priority 1 in output class P and with disposition equals HOLD. tapeaddr = write the account file to a tape unit. The format is: cuu X'cuu' (cuu,X'ss') (X'cuu',X'ss") (cuu,ss) (X'cuu',ss) DISK = write the account file to disk. filename = 1-7 alphameric characters. DEL = delete the account file.

{PALTER
A}
	queue,{jobname[,jobnumber]
	ALL
	*abc
	class1}
	[,PRI=priority]
	[,DISP=disposition]
	[,CLASS=class2]
	[,COPY=number-of-copies]
	[,REMOTE=remid]
		Changes the attribute parameters of a queue entry.
		
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		ALL = alter all queue entries
		*abc = any combination of 1-7 alphameric characters
		class1 = A-Z or 0-4
		priority = 0-9
		disposition = H: hold
		K: keep after processing
		L: leave
		D: delete after processing
		
		class2 = A-Z or 0-4
		number-of-copies = 0-99
		remid = 0-200; 0 = central operator

{PBRDCST
B}
	remid,'text'	Transmits a message from the central operator to the remote user(s).
		
		remid = 0-200 or ALLUSERS;
		0 = central operator
		text = 1-40 characters

{PCANCEL C}	[STATUS]	Terminates printing initiated by a PDISPLAY command.

Figure 29. Summary of Central Installation Operator Commands (Part 1 of 4)

Operation	Operand	Remarks
		Deletes from the specified queue:
{PDELETE	{ queue, {jobname[,jobnumber]	a specific job
L}	ALL	all jobs
	queue, class	a class
	*abc}	all jobs beginning with the letters abc
	MSG[,n] }	queue = LST, PUN, or RDR
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		ALL = delete all queue entries
		class = A-Z or 0-4
		abc = any combination of 1-7 alphameric characters
		n = delete ALLUSERS-type message numbered n. If omitted, all ALLUSERS-type messages are deleted.
		Displays the status in a specified queue of:
{PDISPLAY	{queue,jobname[,jobnumber]	a specific job
D}	queue[,ALL]	all entries of a specific job
	queue,HOLD	all non-dispatchable entries
	queue,FREE	all dispatchable entries
	queue,RJE[,remid]	all RJE-type entries
	queue,LOCAL	all entries submitted by or routed to the central location
	queue,*abc	all jobs beginning with the letters abc
	queue,class	all entries with a specified class
		Displays the status of:
	ALL[,listaddr]	all entries in all queues
	HOLD	all non-dispatchable entries in the system
	FREE	all dispatchable entries in the system
	RJE[,remid]	all RJE-type entries in the system
	LOCAL	all entries in the system submitted by or routed to the central location
	*abc	all jobs in the system beginning with the letters abc
	MSG	all ALLUSERS-type messages
	A	all active reader/writer tasks
	M	all system messages
	Q}	number of free records and free account records
		queue = LST, PUN, or RDR
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		remid = 0-200; 0 = central operator
		*abc = any combination of 1-7 alphameric characters
		class = A-Z or 0-4
		listaddr = cuu or X'cuu'
PEND	[uraddr	Terminates POWER/VS, or
	KILL[,uraddr]]	terminates POWER/VS for emergency reasons.
		uraddr = cuu or X'cuu'

Figure 29. Summary of Central Installation Operator Commands (Part 2 of 4)

Operation	Operand	Remarks

PFLUSH
F
	uraddr[HOLD]
	partition[,HOLD]
		Flushes an active job or partition and, if specified, puts it in the hold state.
		
		uraddr = cuu or X'cuu'
		partition = BG, F4, F3, F2, or F1.

PGO
G
	uraddr
	partition,cuu
		Activates a task or a partition.
		
		uraddr = cuu or X'cuu'
		partition = ~~BG, F1~~, F2, F3, or F4.

PINQUIRE I	lineaddr ALL	Provides status information for one line or for all lines. lineaddr = cuu or X'cuu'
PRELEASE R	queue,jobname[,jobnumber] queue[,ALL] queue,class queue,*abc	Releases from the specified queue and makes them dispatchable: one entry all entries all entries of a specific class all entries beginning with the letters abc queue = LST, PUN, or RDR jobname = 2-8 alphameric characters jobnumber = 1-5 digits class = A-Z, or 0-4 *abc = any combination of 1-7 alphameric characters
PRESTART T	uraddr[,n]	Restarts a list writer task uraddr = cuu or X'cuu' n = 0-9999 (with or without minus sign)

PSETUP	uraddr[,n]	Prints the page layout.
		
		uraddr = cuu or X'cuu'
		n = 1 or 2 digits

PSTART
S
	task,uraddr[,class][,bufno]
	task,uraddr,tapeaddr
	partition[,class][,outputclass]
	RJE,lineaddr[password]
		Starts an RDR, LST, or PUN task, partition or RJE line.
		
		task = RDR, LST, or PUN
		uraddr = cuu or X'cuu'
		bufno = 1 or 2 (card reader only)
		class = A-Z or 0-4
		tapeaddr = X'cuu' (only for list and punch tasks)
		partition = BG, F4, F3, F2, or F1
		outputclass = A-Z
		lineaddr = cuu or X'cuu'
		password = up to 8 alphameric characters

Figure 29. Summary of Central Installation Operator Commands (Part 3 of 4)

Operation	Operand	Remarks
{PSTART} {S} (continued)	RDR,uraddr[,class],uraddr [,bufno]	Starts a reader task that merges a 3540 diskette file with card input.
	RDR,uraddr[,class][,'file-id'] [,file-id]	Starts a reader task that reads both JCL statements and data from a 3540 diskette.
	[,1 [,S][,V]] [,number-of-diskettes]	'file-id' = filename. If not placed between quotes, blanks are allowed number-of-diskettes = maximum number of diskettes to be read (one to three digits, 1-255) S = volume sequence checking V = check on file verification

{PSTOP}
{P}
	{uraddr[,EOJ]
	[,RESTART]
	partition
	lineaddr[,EOJ]}
		Stops a task, partition, or RJE line.
		uraddr = cuu or X'cuu'
		partition = BG, F4, F3, F2, or F1
		lineaddr = cuu or X'cuu'

Figure 29. Summary of Central Installation Operator Commands (Part 4 of 4)

Operation	Operand	Remarks

* .. ALTER	queue, {jobname[,jobnumber]
	ALL
	*abc
	class1}
	[,PRI= priority]
	[,DISP=disposition]
	[,CLASS=class2]
	[,COPY=number-of-copies]
	[,REMOTE=remid]
		Changes the attribute parameters of jobs submitted by or routed to the remote user.
		queue = LST, PUN, or RDR
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		*abc = any combination of from 1-7 alphameric characters
		disposition = H: hold
		K: keep after processing
		L: leave
		D: delete after processing
		
		class2 = A-Z or 0-4
		number-of-copies = 0-99
		remid = 0-200;
		0 = central operator

* .. BRDCST	remid,'text'	Transmits a message to the central operator, to another user, or to all users (ALLUSERS).
		
		remid = 0-200, or ALLUSERS
		0 = central operator
		text = 1-59 characters

* .. DELETE	{queue,jobname[,jobnumber]
	queue,ALL
	queue,class
	queue,*abc
	MSG[,n]}
		Deletes jobs or messages submitted by or routed to the remote user.
		queue = LST, PUN, or RDR
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		class = A-Z or 0-4
		*abc = any combination of 1-7 alphameric characters
		MSG,n where n = message number

* .. DISPLAY	{queue,jobname[,jobnumber]
	queue[,ALL]
	queue,HOLD
	queue,FREE
	queue,*abc
	queue,class
	
	ALL
	HOLD
	FREE
	*abc
	MSG}
		Displays the status of jobs submitted by or routed to the remote user.
		queue = LST, PUN, or RDR
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		*abc = any combination of from 1-7 alphameric characters
		class = A-Z or 0-4

* .. FLUSH	{task task,HOLD}	Flushes an active RJE writer task. task = LST or PUN
* .. GO	task	Reactivates an RJE writer task. task = LST or PUN

Figure 30. Summary of Central Installation Operator Commands (Part 1 of 2)

Operation	Operand	Remarks

* .. RELEASE	queue,jobname[,jobnumber]
	queue[,ALL]
	queue,class
	queue,*abc
		Releases jobs submitted by or routed to the remote user.
		queue = LST, PUN, or RDR
		jobname = 2-8 alphameric characters
		jobnumber = 1-5 digits
		class = A-Z or 0-4
		*abc = any combination of 1-7 alphameric characters

* .. RESTART	task
	task,n
		Restarts an RJE writer task.
		task = LST or PUN
		n = 0-9999

* .. SETUP	LST[,n]	Prints the page layout. n = 1-2 digits
* .. SIGNOFF	(no operand)	Terminates a session by the remote user.

* .. SIGNON	remid[,password]
	[,user-information]
		Starts a session by the remote user.
		
		remid = 1-200
		password = up to 8 alphameric characters
		user-information = up to 16 bytes

* .. START	task
	task,class
	MSG
		Starts an RJE writer task or the printing of messages.
		task = LST or PUN
		class = A-Z

* .. STOP	task task,EOJ task,RESTART MSG	Stops an RJE writer task or the printing of messages. task = LST or PUN

Figure 30. Summary of POWER/VS RJE Terminal Commands (Part 2 of 2)

Bibliography

This publication makes reference to the following publications:

DOS/VS Supervisor and I/O Macros, GC33-5373

DOS/VS Tape Labels, GC33-5374

DOS/VS DASD Labels, GC33-5375

DOS/VS System Generation, GC33-5377

DOS/VS Operating Procedures, GC33-5378

DOS/VS Messages, GC33-5379

DOS/VS Serviceability Aids and Debugging Procedures, GC33-5380

IBM 3211/3216/3811 Component Description and Operating Procedures, GA24-3543

IBM System/370 Model 125 Functional Characteristics, GA33-1506

IBM System/370 Model 115 Functional Characteristics, GA33-1510.

IBM 3340 Fixed Head Feature User's Guide, GA26-1632

DOS/VS and OS/VS TOLTEP for VTAM, GC28-0663

DOS/VS VTAM Network Operating Procedures, GC27-0025.

Index