

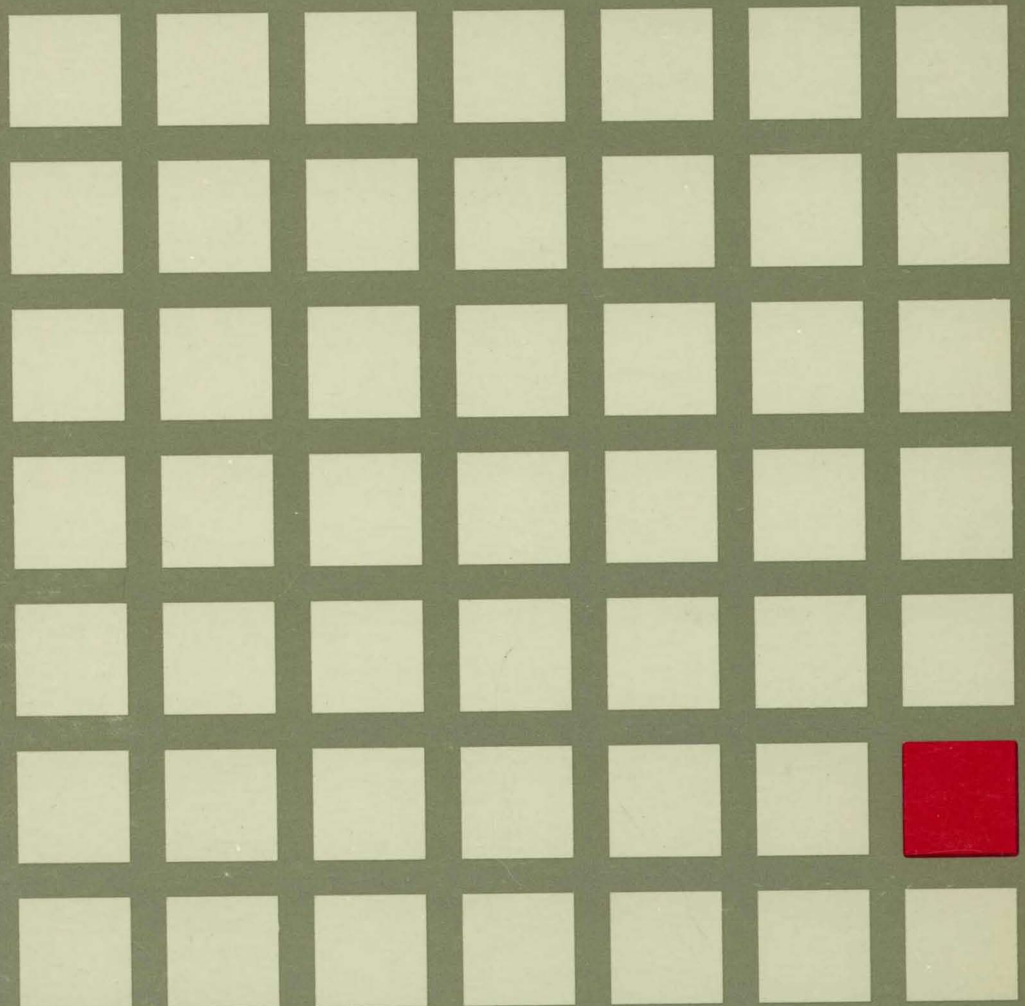


Virtual Machine/System Product

SC19-6209-05

CMS Command Reference

Release 6





Virtual Machine/System Product

SC19-6209-05

CMS Command Reference

Release 6

Sixth Edition (July 1988)

This edition, SC19-6209-05, is a major revision of SC19-6209-04 and applies to Release 6 of the Virtual Machine/System Product (VM/SP), program number 5664-167, and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information contained herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370, 30xx, 4300, and 9370 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

Summary of Changes

For a list of changes, see page 821.

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead.

Ordering Publications

Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. Publications are *not* stocked at the address given below.

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 Corporation, Information Development, Dept. G60, P.O. Box 6, Endicott, NY, U.S.A. 13760. IBM may use or distribute whatever information you supply in anyway it believes appropriate without incurring any obligation to you.

The form for reader's comments provided at the back of this publication may also be used to comment on the VM/SP online HELP facility.

Preface

Use this publication as a reference manual. It contains all of the command formats, syntax rules, and operand and option descriptions for CMS commands for general users.

If you are using Virtual Machine/System Product (VM/SP) for the first time, refer to the *VM/SP CMS Primer*, SC24-5236, for introductory tutorial information about VM/SP. If you are using a line-oriented display terminal, then refer to the *VM/SP CMS Primer for Line-Oriented Terminals*, SC24-5242. The *VM/SP CMS User's Guide*, SC19-6210, contains tutorial information and functional descriptions of CMS commands, as well as information on using the editor and exec facilities of CMS. You should be familiar with the contents of the *VM/SP CMS User's Guide* before you attempt to use this reference manual. For most of the CMS commands described in this publication, you may find additional useful notes in the *VM/SP CMS User's Guide*.

This publication has six chapters:

“Chapter 1. Introduction and General Concepts” describes the components of the VM/SP system and tells you how to enter CMS commands. It lists the notational conventions used in this manual, so that you can interpret the command format descriptions in the following chapters. Chapter 1 also contains information about the CMS command search order and a summary of all the CMS commands available under VM/SP, including those not for general users.

“Chapter 2. CMS Commands” contains complete format descriptions, and operand and option lists, for the CMS commands available to general users. Each command description contains usage notes and lists the responses and error messages (with associated return codes) produced by the command.

“Chapter 3. CMS Commands for Windowing” contains windowing and virtual screen CMS commands to create and modify windows and virtual screens, including Border commands to manage your windows.

“Chapter 4. Special Commands Used in Command Environments” contains complete format descriptions, and operand and option lists, for the special CMS commands available only in command environments, such as FILELIST. Each command description contains usage notes and lists responses and error messages (with associated return codes) produced by the command.

“Chapter 5. HELP and HELPCONV Format Words” describes the formats, operands, and defaults of the HELP facility format words. HELP format words are used in HELP description files when the user wants HELP to format output when the HELP file is processed.

“Chapter 6. System Messages” lists the common system messages and return codes that you might receive when you execute a command.

The chapters that formerly contained the “CMS Functions” and the “CMS Macro Instructions” have been moved to a new publication, the *VM/SP Application Development Reference for CMS*, SC24-5284. Refer to that manual for information on functions that are available and for CMS macro instructions you can use when

you write programs to execute in CMS. There is also a chapter on callable services library (CSL) routines.

The DEBUG command no longer places you in the DEBUG subcommand environment and consequently the chapter on the DEBUG subcommand environment has been removed. See "DEBUG" on page 91 for details on the changes to the DEBUG command.

This publication also has two appendixes:

"Appendix A: VSE/VSAM Functions Not Supported in CMS" lists the restrictions on the use of access method services and VSAM in the CMS/DOS environment of CMS.

"Appendix B: OS/VS Access Method Services and VSAM Functions Not Supported in CMS" lists the restrictions for OS programmers using access method services and VSAM in CMS.

The appendixes on the EDIT subcommands and the EXEC control statements have been removed. Use the HELP command to get information on any of these subcommands or control statements.

Contents

Chapter 1. Introduction and General Concepts	1
CMS Environment	1
Entering CMS Commands	2
Character Set Usage	3
Naming CMS Files	4
Naming Shared File System (SFS) Directories	4
Notational Conventions	6
Pattern Matching	8
CMS Command Search Order	9
CMS Command Execution Characteristics	10
Summary of CMS Commands	11
Chapter 2. CMS Commands	25
ACCESS	27
AMSERV	33
ASSEMBLE	36
ASSGN	43
CATCHECK	47
CMDCALL	49
CMSBATCH	50
CMSSERV	51
COMPARE	53
CONVERT COMMANDS	55
CONWAIT	59
COPYFILE	60
CP	72
CREATE ALIAS	73
CREATE DIRECTORY	77
CREATE LOCK	79
CREATE NAMEDEF	83
CSLLIST	85
DEBUG	91
DEFAULTS	93
DELETE LOCK	96
DELETE NAMEDEF	99
DESBUF	100
DIRLIST	101
DISK	108
DLBL	114
DOSLIB	128
DOSLKED	130
DROPBUF	134
DSERV	135
EDIT	138
ERASE	141
ESERV	147
EXEC	149
EXECDROP	153
EXECIO	155
EXECLOAD	171
EXECMAP	173
EXECOS	176

EXECSTAT	178
EXECUPDT	180
FETCH	183
FILEDEF	186
FILELIST	203
FINIS	219
FORMAT	220
GENDIRT	225
GENMOD	226
GENMSG	233
GLOBAL	237
GLOBALV	240
GRANT AUTHORITY	251
HELP	255
HELPCONV	265
IDENTIFY	267
IMMCMD	269
INCLUDE	271
LABELDEF	277
LISTDIR	282
LISTDS	285
LISTFILE	291
LISTIO	300
LKED	302
LOAD	308
LOADLIB	324
LOADMOD	328
MACLIB	331
MACLIST	335
MAKEBUF	341
MODMAP	342
MOREHELP	343
MOVEFILE	345
NAMEFIND	349
NAMES	358
NETDATA	364
NOTE	376
NUCXDROP	384
NUCXLOAD	385
NUXMAP	389
OPTION	392
OSRUN	394
PARSECMD	395
PEEK	398
PRINT	403
PROGMAP	407
PSERV	410
PUNCH	412
QUERY	415
RDR	454
RDRLIST	459
READCARD	466
RECEIVE	472
RELEASE	481
RELOCATE	484
RENAME	488

RESERVE	493
REVOKE AUTHORITY	496
RSERV	499
RTNDROP	501
RTNLOAD	503
RTNMAP	507
RTNSTATE	511
RUN	514
SEGMENT	517
SEGMENT ASSIGN	518
SEGMENT LOAD	519
SEGMENT PURGE	522
SEGMENT RELEASE	524
SEGMENT RESERVE	525
SENDFILE	527
SENTRIES	536
SET	537
SET ABBREV	538
SET APL	539
SET AUTOREAD	540
SET BLIP	541
SET CMSTYPE	542
SET COMDIR	543
SET DOS	545
SET DOSLNCNT	547
SET DOSPART	548
SET EXECTRAC	550
SET FILEPOOL	551
SET FILEWAIT	553
SET FULLREAD	555
SET IMESCAPE	557
SET IMPCP	558
SET IMPEX	559
SET INPUT	560
SET INSTSEG	561
SET KEYPROTECT	562
SET LANGUAGE	563
SET LDRTBLS	567
SET LINEND	568
SET LOADAREA	569
SET NONDISP	572
SET NONSHARE	573
SET OUTPUT	574
SET PROTECT	575
SET RDYMSG	576
SET REDTYPE	577
SET RELPAGE	578
SET REMOTE	579
SET SERVER	580
SET STORECLR	581
SET SYSNAME	582
SET TEXT	583
SET THRESHOLD	584
SET TRANSLATE	586
SET UPSI	588
SETKEY	589

SETPRT	590
SORT	593
SSERV	595
START	597
STATE/STATEW (ESTATE/ESTATEW)	600
SVCTRACE	602
SYNONYM	606
TAPE	610
TAPEMAC	619
TAPPDS	622
TELL	627
TXTLIB	629
TYPE	633
UPDATE	636
VALIDATE	649
XEDIT	651
XMITMSG	658
Immediate Commands	664
HB	664
HI	665
HO	665
HT	665
HX	666
RO	666
RT	666
SO	667
TE	667
TS	667
Chapter 3. CMS Commands for Windowing	669
ALARM VSCREEN	673
CLEAR VSCREEN	674
CLEAR WINDOW	675
CURSOR VSCREEN	676
DEFINE VSCREEN	679
DEFINE WINDOW	683
DELETE VSCREEN	686
DELETE WINDOW	687
DROP WINDOW	688
GET VSCREEN	690
HIDE WINDOW	692
MAXIMIZE WINDOW	694
MINIMIZE WINDOW	696
POP WINDOW	697
POSITION WINDOW	699
PUT SCREEN	700
PUT VSCREEN	702
QUERY	704
REFRESH	719
RESTORE WINDOW	720
ROUTE	721
SCROLL	724
SET	727
SET APL	728
SET BORDER	729
SET CHARMODE	732

SET CMSPF	734
SET FULLREAD	737
SET FULLSCREEN	739
SET LINEND	748
SET LOCATION	749
SET LOGFILE	750
SET NONDISP	752
SET REMOTE	753
SET RESERVED	754
SET TEXT	756
SET VSCREEN	757
SET WINDOW	759
SET WMPF	761
SHOW WINDOW	764
SIZE WINDOW	766
WAITREAD VSCREEN	767
WAITT VSCREEN	770
WRITE VSCREEN	772
Border Commands	778
B	779
C	779
D	779
F	780
H	780
L	780
M	781
N	781
O	781
P	782
R	782
S	782
X	783
Chapter 4. Special Commands Used in Command Environments	785
ALIALIST	786
AUTHLIST	789
DISCARD	792
EXECUTE	794
Chapter 5. HELP and HELPCONV Format Words	797
.BX (BOX)	798
.CM (COMMENT)	800
.CS (CONDITIONAL SECTION)	801
.FO (FORMAT MODE)	803
.IL (INDENT LINE)	805
.IN (INDENT)	806
.MT (MENU TYPE)	807
.OF (OFFSET)	808
.SP (SPACE LINES)	809
.TR (TRANSLATE CHARACTER)	810
Chapter 6. System Messages	811
Command Syntax Error Messages	811
Shared File System (SFS) Error Messages	813
File Error Messages	814

Appendix A. VSE/VSAM Functions Not Supported in CMS	815
Appendix B. OS/VS Access Method Services and VSAM Functions Not Supported	817
Summary of Changes	821
Glossary of Terms and Abbreviations	831
Bibliography	853
Index	859

Chapter 1. Introduction and General Concepts

Virtual Machine/System Product (VM/SP) is a program product that controls “virtual machines.” A virtual machine is the functional equivalent of a real computer that you control from your terminal, using a command language.

The command languages, which correspond to the components of the VM/SP system:

- The Control Program (CP) controls the resources of the real machine; that is, the physical computer system in your computer room. The CP commands are described in *VM/SP CP General User Command Reference* and *VM/SP CP System Command Reference*.
- The Conversational Monitor System (CMS) is a conversational operating system designed to run under CP. CMS can simulate many of the functions of OS (Operating System) and DOS (Disk Operating System), so that you can run many OS and DOS programs in a conversational environment. This publication describes general use CMS commands that you can use in the CMS environment.
- The Interactive Problem Control System (IPCS) is an interactive, online facility for reporting and diagnosing software failures and for managing problem information and status. VM/SP IPCS can perform these functions for: CP ABEND dumps, CMS, PVM, RSCS, and GCS dumps, and any dump created by the VMDUMP command. IPCS runs in the CMS command environment. For more information, refer to the *VM/SP Interactive Problem Control System Guide and Reference*.

Each of the above components has a unique “command environment” that must be active in order for a command to be accepted. For CMS users, the two basic command environments are the CP command environment and the CMS command environment. By default, CP commands are acceptable input in the CMS command environment; if you enter a CP command, CP executes it, but control returns to the CMS environment. IPCS DUMPSCAN subcommands must be entered from the DUMPSCAN environment.

CMS Environment

The CMS command language allows you to create, modify, and, in general, manipulate a system of files.

The OS/VS Assembler and many OS/VS and VSE (DOS) language processors can be executed under CMS. For example, the OS/VS BASIC, FORTRAN IV (G1), COBOL and PL/I compilers, as well as the DOS PL/I and DOS/VS COBOL compilers, can execute under CMS. You can find a complete list of language processors that can be executed under CMS in the *VM/SP Introduction*. CMS invokes the assembler and the compilers when you issue the appropriate CMS commands. The ASSEMBLE command is described in this manual; the supported compiler commands are described in the appropriate Licensed Program publications.

CMS commands allow you to read cards from a virtual card reader, punch cards to a virtual card punch, and print records on a virtual printer. Many commands are provided to help you manipulate your virtual disks, your directories in the Shared

File System (SFS), and the files that reside on your minidisks and directories. The CMS commands are described in “Chapter 2. CMS Commands.”

A special set of CMS commands becomes available to you when you issue the command:

```
set dos on
```

These commands, called CMS/DOS commands, simulate various functions of the VSE Operating System (DOS) in your CMS virtual machine. When the CMS/DOS environment is active, the CMS/DOS commands are an integral part of the CMS command language; they are listed alphabetically among the other CMS commands in “Chapter 2. CMS Commands.”

In addition, certain CMS commands allow you to use CMS in a full-screen environment. You can display CMS in a window, enter commands from anywhere on the physical screen, scroll through information in windows, and define your own windows. The windowing commands are described in “Chapter 3. CMS Commands for Windowing.”

You may use certain special commands from certain command environments. The function, formats, and operands of these special commands are described in “Chapter 4. Special Commands Used in Command Environments.” The command environments are:

CSLLIST	displays a list of callable services library (CSL) routines
DIRLIST	displays a list of Shared File System (SFS) directories
FILELIST	displays a list of files and SFS directories
MACLIST	displays a list of MACLIB members
RDRLIST	displays a list of files on your virtual reader

The HELP format words are used to create HELP ‘text’ information for user-defined commands, execs, and messages. The function, formats, and operands of the HELP facility format words are described in “Chapter 5. HELP and HELPCONV Format Words.”

Many CMS commands can issue system messages in addition to the messages listed with each command. These system messages are listed in “Chapter 6. System Messages.”

Entering CMS Commands

A CMS command consists of a command name, usually followed by one or more positional operands and, in many cases, by an option list. CMS commands described in this publication are shown in the format:

COMMAND NAME	[operands...] [(options...D)]
---------------------	-------------------------------

You must use one or more blanks to separate each entry in the command line unless otherwise indicated. For an explanation of the special symbols used to describe the command syntax, see “Notational Conventions” on page 6.

Command Name

The command name is an alphameric symbol of one to eight characters. In general, the names are based on verbs or verb-noun combinations that describe the function you want the system to perform. For example, you may want to find out information concerning your CMS files. In this case, you would use either the FILELIST or the LISTFILE command.

Command Operands

The command operands are keywords and/or positional operands of one to eight, and in a few cases, one to seven alphameric characters each. In certain cases there are more than eight characters, and CMS may ignore any characters after the first eight. The operands specify the information on which the system operates when it performs the command function.

You must write the operands in the order in which they appear in the command formats unless otherwise specified. When you are using CMS, blanks may optionally be used to separate the last operand from the option list. CMS recognizes a left parenthesis "(" as the beginning of an option list; it does not have to be preceded by a blank.

Command Options

The command options are keywords used to control the execution of the command. The command formats in this publication show all the options for each CMS command.

The option list must be preceded by a left parenthesis; the closing parenthesis is not necessary.

Character Set Usage

CMS commands may be entered using a combination of characters from six different character sets. The contents of each of the character sets is shown in Table 1.

Character Set	Names	Symbols
Separator	Blank	
National	Dollar Sign Pound Sign At Sign	\$ # @
Alphabetic	Uppercase Lowercase	A - Z a-z
Numeric	Numeric	0 - 9
Alphameric	National Alphabetic Numeric	\$, #, @ A - Z a - z 0 - 9
Special		All other characters

Naming CMS Files

When you create a CMS file, you can give it any file name and file type you wish. The rules for forming file names and file types are:

- The file name and file type can each be from one to eight characters.
- The valid characters are A-Z, a-z, 0-9, \$, #, @, +, - (hyphen), : (colon), and _ (underscore).

Note: Lowercase letters within a file ID are valid for use within the CMS file system. However, some CMS commands do not support file IDs that contain lowercase letters.

Naming Shared File System (SFS) Directories

When you create an SFS directory you can give it any name you want; however, there are a few rules governing directory names:

- The complete directory name (also referred to as *dirname*) is made up of its parent directory's name plus any subdirectory names separated by a period.
- A directory name can be from one to sixteen characters.
- The first character must be alphabetic, and the remaining characters can be alphabetic or numeric.
- Two or more subdirectories may have the same name as long as each of the subdirectories has a different parent directory.

This example illustrates the above rules.

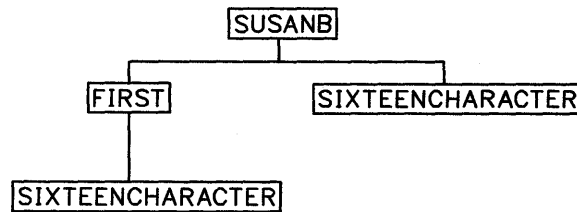


Figure 1. Example of a directory structure

Given that this directory structure exists in a file pool named FPOOL1 and your user ID is SUSANB, the name of each of the four directories is:

```

FPOOL1:SUSANB
FPOOL1:SUSANB.FIRST
FPOOL1:SUSANB.SIXTEENCHARACTER
FPOOL1:SUSANB.FIRST.SIXTEENCHARACTER
  
```

The directory identifier is abbreviated as *dirid* throughout the rest of this book. It would be cumbersome to enter the complete directory name every time you use *dirid* in a command, so a convenient notation is allowed. There are four ways that *dirid* can be represented:

$$dirid = \left\{ \begin{array}{l} fm \\ +fm.ni [.ni+1 . .ni+7] \\ -fm [.ni.ni+1 . .ni+7] \\ [filepoolid:][userid].[n1.n2. .n8] \end{array} \right\}$$

Figure 2. Dirid definition

where:

fm

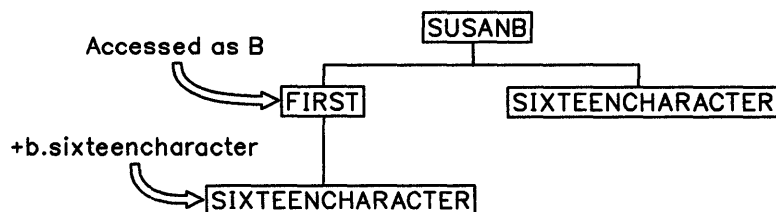
is the file mode letter assigned to the directory when it is accessed. To access directories, use the ACCESS command.

Note: It is important to remember a few rules about file mode numbers when you are specifying *fm*. Generally, you do not want to use file mode numbers unless you are specifying a file or set of files. Since file mode numbers are attributes of files, and not SFS directories or minidisks, you cannot specify file mode numbers on commands that operate on an entire minidisk or SFS directory. For example, do not use file mode numbers on such commands as ACCESS, DIRLIST, FORMAT, RELOCATE, and RELEASE unless you are specifying a set of files.

$+fm.ni [.ni+1 . .ni+7]$

represents the name of a directory. The $+fm$ means to start at the directory accessed as *fm* and go down the directory structure. The lower level directories (subdirectories) are *ni* through *ni+7*, and together they make up a logical path through the directory structure. The total number of subdirectories including all levels to get to the level above *fm*, and then adding all *ni* directories cannot be more than 8. An example of this type of *dirid*, where the directory FPOOL1:SUSANB.FIRST is accessed as B, is:

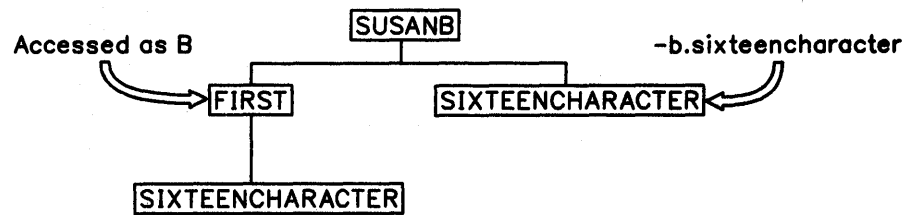
$+b.sixteencharacter$



$-fm [.ni.ni+1 . .ni+7]$

represents the name of a directory. The $-fm$ means to start at the directory accessed as *fm* and back up one level to its parent directory. The *ni* through *ni+7* are lower level directories (subdirectories), as described above. An example of this type of *dirid* is:

$-b.sixteencharacter$



`[filepoolid:][userid].[n1.n2. . n8]`

represents the full name of a directory. This form of *dirid* is also referred to as the *dirname*.

filepoolid:

is the one to eight character name of the file pool that the specified directory resides in. The *filepoolid* must always end with a colon (:). If *filepoolid:* is not specified, the default file pool is used.

userid

is the user ID of the owner of the directory and the name of the user's top directory. If *userid* is not specified, your user ID is assumed.

. (period)

is a separator between *userid* and subdirectory names or a separator between subdirectory names. If the period is used by itself, it indicates your top directory in your default file pool. For example, issuing:

```
access . a
```

accesses your top directory as A.

n1.n2...n8

are the directory levels. *userid* is the top directory and *n1* through *n8* are subdirectories. The first letter must be alphabetic.

Notational Conventions

The notation used to define the command syntax in this publication is:

- Truncations and Abbreviations of Commands

Where truncation of a command name is permitted, the shortest acceptable version of the command is represented by uppercase letters. (Remember, however, that CMS commands can be entered with any combination of uppercase and lowercase letters.) The following example shows the format specification for the FILEDEF command.

FILEdef

This format means that **FI**, **FIL**, **FILE**, **FILED**, **FILEDE**, and **FILEDEF** are all valid specifications for this command name.

Operands and options are specified in the same manner. Where truncation is permitted, the shortest acceptable version of the operand or option is represented by uppercase letters in the command format box. If no minimum truncation is noted, the entire word (represented by all uppercase letters) must be entered.

Abbreviations are shorter forms of command operands and options. Abbreviations for operands and options are shown in the description of the individual operands and options that follow the format box. For example, the abbreviation for DISK in the ASSEMBLE command is DI. Only these two forms are valid and no truncations are allowed. The format box contains

DISK

and the description that follows the format box is

DISK**DI**

- Keywords, such as command names, operands, and options, appear in the format box and parameter descriptions in **bold** letters.
- Lowercase letters, words, and symbols that appear in the command format box in *italics* represent variables that you will substitute with specific information. For example,

fn indicates that you should enter a file name with the command.

- Uppercase letters and words, and the following symbols, should be entered as specified in the format box.

asterisk	*
colon	:
comma	,
equal sign	=
hyphen	-
parentheses	()
period	.

- The abbreviations *fn*, *ft*, and *fm* refer to file name, file type, and file mode, respectively. The combination *fn ft [fm]* is also called the file identifier or file ID.

When a command format box shows the characters *fn ft fm* or *fileid* and they are not enclosed by brackets or braces, it indicates that a CMS file identifier must be entered. If an asterisk (*) appears beneath *fn*, *ft*, or *fm*, it indicates that an asterisk may be coded in that position of the file ID. The operand description describes the usage of the *.

- Choices are represented in the command format boxes by stacking.

A
B
C

- An underscore indicates an assumed default option. If an underscored choice is selected, it need not be specified when the command is entered.

For example, the representation:

A
B
C

indicates that either A, B, or C may be selected. However, if B is selected, it need not be specified. Or, if none is entered, B is assumed.

- The use of braces denotes choices, one of which *must* be selected.

Example: The representation:

{ A
B
C }

indicates that you *must* specify either A, or B, or C. If a list of choices is enclosed by neither brackets nor braces, it is to be treated as if enclosed by braces.

- The use of brackets denotes choices, one of which *may* be selected.

Example: The representation:

[A
B
C]

indicates that you may enter A, B, or C, or you may omit the field.

- In instances where there are nested braces or brackets on the text lines, the following rule applies: nested operand selection is dependent upon the selection of the operand of a higher level of nesting.

Example:

Level 1 Level 2 Level 3
[filename [filetype [filemode]]]

where the highest level (Level 1) of nesting is the operand that is enclosed in only one pair of brackets and the lowest level (Level 3) of nesting is the operand that is enclosed by the maximum number of brackets. Thus, in the previous example, the user has the option of selecting a file by *filename* only or *filename filetype* only or by *filename filetype filemode*. The user cannot select file type alone because file type is nested within file name and our rule states: the higher level of nesting must be selected in order to select the next level (lower level) operand. The same is true if the user wants to select file mode; file name and file type must also be selected.

- An ellipsis indicates that the preceding item or group of items may be repeated more than once in succession.

Example: The representation:

(options...)

indicates that more than one option may be coded within the parentheses.

Pattern Matching

A few CMS commands allow you to use two special characters, * (asterisk) and % (percent), in the *fn* and *ft* operands, if you want to specify a subset of your files rather than just one file.

These commands are:

- CREATE ALIAS
- CREATE LOCK
- DELETE LOCK
- FILELIST
- GRANT AUTHORITY

- LISTFILE
- QUERY ALIAS
- QUERY AUTHORITY
- QUERY LOCK
- RELOCATE
- REVOKE AUTHORITY

The two special characters, * (asterisk) and % (percent), have the following meanings:

- * represents any number of character(s). As many asterisks as required can appear *anywhere* in a file name or file type. However, the total number of characters, including the asterisks, may not exceed eight.

For example, if you enter:

```
filelist *d* *file*
```

you are requesting that the list contain all files on your disk or directory accessed as A whose file name contains “d” and whose file type contains “file.”

The list might contain the following files:

```
D      FILE      A1
YOURDATA AFILE1  A1
HISDATA AFILE2  A1
ADOG    1DOGFIL  A2
```

- % is a place holding character that means a *single* character, but any character will do. As many percent symbols as necessary may appear anywhere in a file name or file type. For example, if you enter:

```
filelist %%% stock
```

you are requesting that the list contain all files on your disk or directory accessed as A whose file name is three characters in length and whose file type is “stock.” The list might contain the following files:

```
THE  STOCK  A1
HIS  STOCK  A1
HER  STOCK  A1
```

CMS Command Search Order

When you enter a command line in the CMS environment, CMS has to locate the command to execute. If you have EXEC or MODULE files on any of your accessed disks or directories, CMS treats them as commands; they are known as user-written commands. For CMS to find an EXEC or MODULE file in an SFS directory, you must have read or write authority to the file.

As soon as the command name is found, the search stops and the command is executed. The search order is:

1. Search for an exec with the specified command name:
 - a. Search for an exec in storage. If an exec with this name is found, CMS determines whether the exec has a USER, SYSTEM, or SHARED attribute. If the exec has the USER or SYSTEM attribute, it is executed.

If the exec has the SHARED attribute, the INSTSEG setting of the SET command is checked. When INSTSEG is ON, all accessed disks and directories are searched and the access mode of the CMS installation saved

segment is compared to the mode of an exec with the name that resides on the disk or directory. If the access mode of a saved segment is equal to or higher than the file mode, the exec in that saved segment is executed. Otherwise, the exec on the disk or directory is executed.

- b. Search for a file with the specified command name and a file type EXEC on any currently accessed disk or directory. CMS uses the standard search order (A through Z.) The table of active (open) files is searched first. An open file may be used ahead of a file that resides on a disk or directory earlier in the search order.
2. Search for a translation or synonym of the specified command name. If found, search for an exec with the valid translation or synonym by repeating Step 1.
 3. Search for a module with the specified command name:
 - a. Search for a nucleus extension module.
 - b. Search for a module in the transient area.
 - c. Search for a nucleus resident module.
 - d. Search for a file with file type MODULE on any currently accessed minidisk or directory. The table of active (open) CMS files is searched first. An open file may be used ahead of a file that resides on a minidisk or directory earlier in the search order. The table of active (open) files is searched first. An open file may be used ahead of a file that resides on a disk or directory earlier in the search order.
 4. Search for a translation or synonym of the specified command name. If found, search for a module with the valid translation or synonym by repeating Step 3.

When CMS searches for a translation or synonym (as in Step #'s 2 and 4 above), the translation and synonym tables are searched in the following order:

1. User National Language Translation Table
2. System National Language Translation Table
3. User National Language Translation Synonym Table
4. System National Language Translation Synonym Table
5. CMS User Synonym Table
6. CMS System Synonym Table

Note: For information about the National Language Translation tables (items 1-4), see the SET TRANSLATE command. For information about User and System Synonym tables (items 5-6), see the SYNONYM command.

If the command is not known to CMS, it is passed to CP for execution.

CMS Command Execution Characteristics

Following is an alphabetical list of the CMS commands which require special consideration when called from a user program. For example, a program running in the user area (the storage available to the user) cannot call a CMS command which also runs in the user area. To avoid conflicts with non-relocatable CMS commands, you should ensure that your user programs are relocatable.

Any commands which are listed in this book, but are not in this table are nucleus resident and will not interfere with the execution of a user program.

The **Code** column indicates the execution characteristics of the command.

Code	Meaning
E	indicates that this command is an exec. It may execute one or more CMS commands that run in user free storage or the transient area.
T	indicates that this command executes in the transient area. The transient area is the storage area used for temporary storage of programs or routines.
U	indicates that this command executes in user free storage. All OS free storage pointers are reset.

Command	Code	Command	Code	Command	Code
ALIALIST	E	FCOBOL	E	OSRUN	U
AMSERV	U	FILELIST	E	PEEK	E
ASSEMBLE	U	FORMAT	U	PSERV	U
ASSGN	T	GENDIRT	T	PUNCH	T
AUTHLIST	E	GENMSG	U	RDR	T
CATCHECK	U	GLOBAL	T	RDRLIST	E
CMSBATCH	U	HELPCONV	T	READCARD	T
CMSSERV	E	IOCP	U	RECEIVE	E
COMPARE	T	LABELDEF	T	RESERVE	T
CONVERT COMMANDS	E	LANGGEN	E	RSERV	U
CSLGEN	E	LANGMERG	E	RUN	E
CSLLIST	E	LISTDS	U	SENDFILE	E
DDR	U	LISTIO	T	SETPRT	T
DEFAULTS	E	LKED	U	SORT	U
DIRLIST	E	LOADLIB	U	SSERV	U
DISCARD	E	MACLIB	U	SVCTRACE	T
DOSLIB	U	MACLIST	E	SYNONYM	T
DOSLKED	U	MODMAP	T	TAPE	T
DOSPLI	E	MOREHELP	E	TAPEMAC	U
DSERV	U	MOVEFILE	U	TAPPDS	U
EDIT	E	NAMES	E	TELL	E
ESERV	E	NOTE	E	TXTLIB	U
EXECMAP	T	NUCXDROP	T	TYPE	T
EXECUPDT	E	OPTION	T	UPDATE	U

Summary of CMS Commands

The following three tables contain alphabetical lists of the CMS commands and the functions performed by each. Table 3 and Table 4 lists those commands that are described in this manual. Table 5 lists CMS commands that are described in other publications.

You can enter CMS commands when you are running CMS in your virtual machine, the terminal is idle, and the virtual machine can accept input. However, if CMS is processing a previously entered command and your typewriter terminal keyboard is locked, you must signal your virtual machine via an attention interruption. The

system acknowledges the interruption by unlocking the keyboard. Now you can enter commands.

If your terminal is a display device, there is no problem of entering commands while the virtual machine is busy because its keyboard remains unlocked for additional command input. Note that in these circumstances the command you enter is stacked in the terminal input buffer and is not executed until the command that is currently being executed completes. If more commands are entered than CP can handle, a NOT ACCEPTED message is displayed at the display terminal.

In addition to the commands listed in Table 3, Table 4, and Table 5, there are ten Immediate commands that are handled in a different manner from the others.

Immediate commands may be entered while another command is being executed by pressing the Attention key (or its equivalent), and they are executed immediately. The Immediate commands are:

- HB** Halt batch execution
- HI** Halt Interpretation
- HO** Halt tracing
- HT** Halt typing
- HX** Halt execution
- RO** Resume tracing
- RT** Resume typing
- SO** Suspend tracing
- TE** Trace end
- TS** Trace start

You can define your own immediate commands by using any of the following:

- The IMMCMD macro in an assembler language program
- The IMMCMD command within an exec (CMS EXEC, EXEC 2, System Product Interpreter)
- NUCXLOAD command with the IMMCMD option specified.

CMS Commands Described in the CMS Command Reference

Table 3 (Page 1 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
ACCESS	Allows you to access a minidisk or an SFS directory with a file mode letter.
ALARM VSCREEN	Sounds the terminal alarm the next time the display is refreshed.
AMSERV	Uses access method services utility functions to create, alter, list, copy, delete, import, or export VSAM catalogs and data sets.
ASSEMBLE	Assembles assembler language source code.

Table 3 (Page 2 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
ASSGN	Assigns or unassigns a CMS/DOS system or programmer logical unit for a virtual I/O device.
CATCHECK	Allows a CMS VSAM user (with or without DOS set ON) to use the VSE/VSAM Catalog Check Service Aid to verify a complete catalog structure.
CLEAR VSCREEN	Erases data in the virtual screen by overwriting the data buffer with nulls.
CLEAR WINDOW	Scrolls past all data in the virtual screen to which the window is connected so that no data is displayed in the data area of the window.
CMDCALL	Converts EXEC 2 extended plist function calls to CMS extended plist command calls.
CMSBATCH	Calls the CMS batch facility.
CMSSERV	Starts IBM Enhanced Connectivity Facilities communications between your VM/SP host system and your workstation (IBM Personal Computer).
COMPARE	Compares records in CMS files.
CONVERT COMMANDS	Converts a CMS file containing Definition Language for Command Syntax (DLCS) statements into an internal form for the parsing facility.
CONWAIT	Causes a program to wait until all pending terminal I/O is complete.
COPYFILE	Copies CMS files from one minidisk to another, one SFS directory to another, or between minidisks and directories.
CP	Enters CP commands from the CMS environment.
CREATE ALIAS	Places an additional name for a file in a specified directory.
CREATE DIRECTORY	Creates an SFS directory.
CREATE LOCK	Creates an explicit lock on an SFS directory or a file in an SFS directory.
CREATE NAMEDEF	Assigns a temporary name which can be used by a program, instead of a file name and file type or a fully-qualified directory name.
CSLLIST	Lists information about all members of a specified callable services library.
CURSOR VSCREEN	Positions the cursor on specified line and column in a virtual screen.
DEBUG	Displays state of virtual machine at time ofabend.
DEFAULTS	Sets or displays default options for various commands.
DEFINE VSCREEN	Creates a virtual screen.
DEFINE WINDOW	Creates a window.
DELETE LOCK	Releases the explicit lock placed on a directory or a file in a directory by the CREATE LOCK command.

Table 3 (Page 3 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
DELETE NAMEDEF	Deletes the temporary name given to a file or directory by the CREATE NAMEDEF command, and makes it no longer usable by a program.
DELETE VSCREEN	Removes a virtual screen definition.
DELETE WINDOW	Removes a window definition.
DESBUF	Clears the program stack and the terminal input buffers.
DIRLIST	Lists directories of a specified directory structure in a full screen environment.
DISK	Performs disk-to-card and card-to-disk operations for CMS files. Can be used for files residing on minidisk or in directories.
DLBL	Defines a VSE file name or VSAM <i>ddname</i> and relates that name to a file.
DOSLIB	Deletes, compacts, or lists information about the phases of a CMS/DOS phase library.
DOSLKED	Link-edits CMS text decks or object modules from a VSE relocatable library and places them in executable form in a CMS/DOS phase library.
DROPBUF	Eliminates a program stack buffer.
DROP WINDOW	Moves a window down in the order of displayed windows.
DSERV	Displays information contained in the VSE core image, relocatable, source, procedure, and transient directories.
EDIT	Uses the VM/SP System Product Editor in CMS editor (EDIT) compatibility mode to create or modify a file residing on a minidisk or in an SFS directory.
ERASE	Deletes CMS files from a minidisk, or deletes both files and subdirectories from an SFS directory.
ESERV	Displays, punches or prints an edited (compressed) macro from a VSE source statement library (E sublibrary).
EXEC	Executes special procedures made up of frequently used sequences of commands.
EXECDROP	Purges storage-resident execs.
EXECIO	Does I/O operations between a device and the program stack or a variable.
EXECLOAD	Loads execs into storage.
EXECMAP	Lists storage-resident execs and displays execs in saved segments.
EXECOS	Resets the OS and VSAM environments under CMS without returning to the interactive environment.
EXECSTAT	Obtains status of a specific exec.
EXECUPDT	Produces an updated executable version of a System Product Interpreter source program.

Table 3 (Page 4 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
FETCH	Fetches a CMS/DOS or VSE executable phase.
FILEDEF	Defines an OS <i>ddname</i> and relates that <i>ddname</i> to any device supported by CMS or to a file residing on a minidisk or in an SFS directory.
FILELIST	Lists information about CMS files on a minidisk or in an SFS directory, with the ability to edit and issue commands from the list.
FINIS	Closes an open file on a minidisk or in a directory.
FORMAT	Prepares minidisks in CMS fixed block format.
GENDIRT	Fills in auxiliary module directories.
GENMOD	Generates nonrelocatable CMS files (MODULE files).
GENMSG	Converts a message repository file into an internal form.
GET VSCREEN	Writes data from a CMS file to the specified virtual screen.
GLOBAL	Identifies specific CMS libraries to be searched for macros, copy files, missing subroutines, LOADLIB modules, or DOS executable phases.
GLOBALV	Sets, maintains and retrieves a collection of named variables.
GRANT AUTHORITY	Authorizes other users to read and modify your SFS directories or the files in the directories.
HELP	Displays information about CP, CMS, or user commands, EDIT or XEDIT subcommands, EXEC, EXEC 2 and System Product Interpreter control statements, and descriptions of CMS and CP messages.
HELPCONV	Converts a CMS file into an acceptable form to be used by the HELP facility.
HIDE WINDOW	Prevents the specified window from being displayed, and connects the window to a virtual screen.
IDENTIFY	Displays or stacks user ID, node ID, RSCS ID, date, time, time zone, and day of the week.
IMMCMD	Establishes or cancels Immediate commands from within an exec.
INCLUDE	Brings additional TEXT files into storage and establish linkages.
LABELDEF	Specifies standard HDR1 and EOF1 tape label description information for CMS, CMS/DOS, and OS simulation.
LISTDIR	Lists directories in a specified directory structure.
LISTDS	Lists information about data sets and space allocation on OS, DOS, and VSAM minidisks.
LISTFILE	Lists information about CMS files stored on a minidisk or in an SFS directory.
LISTIO	Displays information concerning CMS/DOS system and programmer logical units.
LKED	Link-edits a CMS TEXT file or OS object module into a CMS LOADLIB.

Introduction

Table 3 (Page 5 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
LOAD	Brings TEXT files into storage for execution.
LOADLIB	Maintains CMS LOADLIB libraries.
LOADMOD	Brings a single MODULE file into storage.
MACLIB	Creates or modifies CMS macro libraries.
MACLIST	Lists information about all members in a specified maclib, with the ability to edit and issue commands from the list.
MAKEBUF	Creates a new program stack buffer.
MAXIMIZE WINDOW	Expands a window to the physical screen size.
MINIMIZE WINDOW	Reduces the size of the window to one line.
MODMAP	Displays the load map of a MODULE file.
MOREHELP	Obtains either additional or related information about the latest valid HELP command you issued.
MOVEFILE	Moves data from one device to another device of the same or a different type.
NAMEFIND	Displays/stacks information from a NAMES file. (NAMES file = file name of <i>userid</i> and file type of NAMES).
NAMES	Displays a menu to create, display or modify entries in a 'userid NAMES' file. (The menu is available only on display terminals.)
NETDATA	Used from an exec to query, receive or send files to users at a network node or on your system.
NOTE	Prepares a 'note' for one or more computer users, to be sent via the SENDFILE command.
NUCXDROP	Deletes specified nucleus extensions.
NUCXLOAD	Loads a nucleus extension.
NUCXMAP	Identifies existing nucleus extensions, including those residing in saved segments.
OPTION	Changes the DOS/VS COBOL compiler (FCOBOL) options that are in effect for the current terminal session.
OSRUN	Loads, relocates, and executes a load module from a CMS LOADLIB or OS module library.
PARSECMD	Calls the parsing facility from within an exec.
PEEK	Displays a file that is in your virtual reader without reading it onto a disk or directory.
POP WINDOW	Moves a window up in the order of displayed windows.
POSITION WINDOW	Changes the location of a window on the physical screen.
PRINT	Spools a specified CMS file to the virtual printer.
PROGMAP	Displays or places on the program stack information on programs currently loaded in storage or in a saved segment.

Table 3 (Page 6 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
PSERV	Copies a procedure from the VSE procedure library onto a CMS minidisk or an SFS directory, displays the procedure at the terminal, or spools the procedure to the virtual punch or printer.
PUNCH	Spools a copy of a CMS file to the virtual punch.
PUT SCREEN	Makes a copy of the physical screen and writes the image to a CMS file.
PUT VSCREEN	Writes the data from the data area of a virtual screen to a CMS file.
QUERY	Requests information about CMS files, minidisks or Shared File System (SFS) directories or other virtual machine characteristics.
RDR	Generates a return code and either displays or stacks a message that identifies the characteristics of the next file in your virtual reader.
RDRLIST	Displays information about files in your virtual reader with the ability to issue commands from the list.
READCARD	Reads data from spooled card input device.
RECEIVE	Reads to your SFS directory or minidisk a file or note that is in your virtual reader.
REFRESH	Updates virtual screens and their associated windows.
RELEASE	Releases an SFS directory or minidisk previously accessed.
RELOCATE	Moves a file or directory structure from one directory to another that you own within the same file pool.
RENAME	Changes the name of a CMS file or directory.
RESERVE	Allocates all available blocks of a 512-, 1K-, 2K-, or 4K-byte block formatted minidisk to a unique CMS file.
RESTORE WINDOW	Returns a maximized or minimized window to its size and location prior to the maximize or minimize.
REVOKE AUTHORITY	Cancels authorities that you granted for a directory or files in a directory.
ROUTE	Directs data of a particular message class to a virtual screen.
RSERV	Copies a VSE relocatable module onto a CMS minidisk or SFS directory, displays it at the terminal, or spools a copy to the virtual punch or printer.
RTNDROP	Cancels the binding of a callable services library routine.
RTNLOAD	Searches for, loads, and binds a callable services library routine to a fixed location in storage, and makes it available for invocation.
RTNMAP	Displays information about the callable services library routines that are currently loaded and bound to an address.
RTNSTATE	Obtains the status of one or more specific callable services library routines.
RUN	Initiates series of functions to be performed on a source, MODULE, TEXT, or EXEC file.

Introduction

Table 3 (Page 7 of 8). CMS Commands Described in CMS Command Reference	
Command	Usage
SCROLL	Moves a window to a new location on the virtual screen.
SEGMENT ASSIGN	Indicates the logical segment to be associated with the physical segment.
SEGMENT LOAD	Loads a saved segment.
SEGMENT PURGE	Purges a saved segment.
SEGMENT RELEASE	Releases the storage held by a segment space.
SEGMENT RESERVE	Creates a segment space for subsequent loading.
SENDFILE	Sends files or notes to one or more computer users, locally or remotely attached, by issuing the command or by using a menu (display terminal only.)
SENTRIES	Determines the number of lines currently in the program stack.
SET	Establishes, sets, or resets CMS virtual machine characteristics.
SETKEY	Changes settings for CMS storage keys.
SETPRT	Loads a virtual 3800 printer.
SHOW WINDOW	Places a window on top of all other displayed windows and connects the window to a virtual screen.
SIZE WINDOW	Changes the number of lines and columns for a specified window.
SORT	Arranges a specified file in ascending order according to sort fields in the data records.
SSERV	Copies a VSE source statement book onto a CMS minidisk or SFS directory, displays it at the terminal, or spools a copy to the virtual punch or printer.
START	Begins execution of programs previously loaded (OS and CMS) or fetched (CMS/DOS).
STATE	Verifies the existence of a CMS file on an accessed minidisk or in an SFS directory.
STATEW	Verifies a file on a read/write file mode.
SVCTRACE	Records information about supervisor calls.
SYNONYM	Uses a table containing synonyms you have created for CMS and user-written commands.
TAPE	Transfers files between tapes and minidisks (or SFS directories), positions tapes, and displays or writes VOL1 labels.
TAPEMAC	Creates CMS MACLIB libraries directly from an IEHMOVE-created partitioned data set on tape.
TAPPDS	Loads OS partitioned data set (PDS) files or card image files from tape to minidisk or SFS directory.
TELL	Sends a message to one or more computer users who are logged on to your computer or to one attached to yours via RSCS.
TXTLIB	Generates and modifies text libraries.

Command	Usage
TYPE	Displays all or part of a CMS file at the terminal.
UPDATE	Makes changes in a program source file as defined by control cards in a control file.
VALIDATE	Verifies the syntax of a file identifier and verifies whether or not a file mode is accessed.
WAITREAD VSCREEN	Used from an exec to update the virtual screen with data, refresh the physical screen, and wait for the next attention interrupt.
WAITT VSCREEN	Updates the virtual screen with data.
WRITE VSCREEN	Enters information in a virtual screen.
XEDIT	Uses the VM/SP System Product Editor to create or modify a CMS file.
XMITMSG	Retrieves a message from a CMS message repository file or your own message repository file.

Special CMS Commands from Command Environments

Command	Usage
AUHLIST	Displays authority information. May be issued only from the DIRLIST or FILELIST environments.
ALIALIST	Displays alias information. May be issued only from the FILELIST environment.
DISCARD	Erases a file (or SFS directory, or MACLIB member) that is displayed from the DIRLIST, FILELIST, MACLIST, RDRLIST, or PEEK command environments.
EXECUTE	Use to issue commands (or execs) from the CSLLIST, DIRLIST, FILELIST, MACLIST, and RDRLIST environments.

CMS Commands Described in other Publications

For those commands not described in this manual, the **Book Code** column indicates the publication that describes the command:

Book Code	Meaning
ADGCMS	Indicates that this command is described in <i>VM/SP Application Development Guide for CMS</i> .
ADMIN	Indicates that this command is described in <i>VM/SP Administration</i> .
EREP	Indicates that this command is described in <i>OS/VSE Environmental Recording Editing and Printing (EREP) Program</i> .
INST	Indicates that this command is described in the <i>VM/SP Installation Guide</i> .

- IOCP UG** Indicates that this command is described in the *Input/Output Configuration Program User's Guide and Reference*.
- IPCS** Indicates that this command is described in the *VM/SP Interactive Problem Control System Guide and Reference*
- OS PP** Indicates that this command invokes an OS program product, available from IBM for a license fee.
- PLNGDE** Indicates that this command is described in the *VM/SP Planning Guide and Reference*.
- SERV** Indicates that this command is described in the *VM/SP Service Guide*.
- SFPROG** Indicates that this command is described in *VM/SP System Facilities for Programming*.
- SFSAD** Indicates that this command is described in *VM/SP CMS Shared File System Administration*.
- VSE PP** Indicates that this command invokes a VSE Program Product, available from IBM for a license fee.

Command	Book Code	Usage
ASM3705	INST	Assembles 3704 3705 Control Program source code.
ASMGEND	SERV	Builds the VM/370 system assembler.
CMSGEND	SERV	Builds CMS command modules and load libraries (LOADLIBs).
CPEREP	EREP	Formats and edits system error records for output.
CSLGEN	ADGCMS	Builds a callable services library from control files, text files and template files.
DCSSGEN	INST	Builds the CMS installation saved segment (CMSINST).
DDR	SFPROG	Performs backup, restores, and copies operations for entire DASD volumes or minidisks.
DELETE ADMINISTRATOR	SFSAD	Removes administrator authority for the specified SFS file pool, from the specified user ID. (For use by file pool administrator only.)
DELETE LOCK	SFSAD	Releases an explicit lock on a file or SFS directory.
DELETE PUBLIC	SFSAD	Removes the connect authority given to public on the ENROLL PUBLIC command. (For use by file pool administrator only.)
DELETE USER	SFSAD	Removes a user from a SFS file pool. (For use by file pool administrator only.)
DIRECT	PLNGDE	Processes the control statements in a CP directory file.
DISKMAP	PLNGDE	Summarizes the MDISK statements in the CP directory file.
DOSGEN	INST	Builds the CMSDOS physical saved segment.
DOSPLI	VSE PP	Compiles DOS PL/I source code under CMS/DOS.

Table 5 (Page 2 of 4). CMS Commands Described in other Publications		
Command	Book Code	Usage
ENROLL ADMINISTRATOR	SFSAD	Adds a file system administrator to the specified SFS file pool. (For use by file pool administrator only).
ENROLL PUBLIC	SFSAD	Gives connect authority for a SFS file pool to all users. (For use by file pool administrator only.)
ENROLL USER	SFSAD	Enrolls a user in the specified SFS file pool. (For use by file pool administrator only.)
EXPAND	SERV	Adds space to a program in object deck form.
FCOBOL	VSE PP	Compiles DOS/VS COBOL source code under CMS/DOS.
FILEPOOL BACKUP	SFSAD	Backs up all data in a user storage group and all associated file pool catalog data.
FILEPOOL CLEANUP	SFSAD	Corrects storage group or administration machine problems.
FILEPOOL FORMAT AUDIT	SFSAD	Formats the security audit data created by file pool server processing.
FILEPOOL RESTORE	SFSAD	Loads the specified file pool storage group from a copy of the storage group data created by FILEPOOL BACKUP processing.
FILESERV BACKUP	SFSAD	Starts a file pool server in dedicated maintenance mode to back up the <i>control data</i> .
FILESERV DEFAUDIT	SFSAD	Adds, changes, or deletes the assignment of the security audit output file for a file pool.
FILESERV DEFBACKUP	SFSAD	Adds, changes, or deletes the assignment of the control data backup file for the file pool.
FILESERV GENERATE	SFSAD	Defines and initializes a new CMS SFS file pool.
IPCSSCAN	IPCS	Provides interactive analysis of dumps and CPTRAP files.
GEN3705	INST	Generates an exec file that assembles and link-edits the 3704 3705 Control Program.
GENSERVE	SERV	Builds CMS Shared File System (SFS) file pool server load modules.
GENTSAF	SERV	Builds the RUNTSAF module and creates a TSAF load map.
GROUP	INST	Builds a GCS configuration file.
IOCP	IOCP UG	Uses the Input/Output Configuration Program
ITASK	INST	Performs most of the installation procedure by invoking other execs and commands.
LANGGEN	SFPROG	Saves all text files for a language in a saved segment, and/or save CP message repository.

Introduction

Table 5 (Page 3 of 4). CMS Commands Described in other Publications		
Command	Book Code	Usage
LANGMERG	SFPROG	Combines all language-related files for an application into one text file.
MODIFY USER	SFSAD	Modifies a user's file space allocation in the Shared File System. (For use by file pool administrator only.)
PRB	IPCS	Update IPCS problem status.
PRELOAD	SERV	Collects multiple text files and reformats them into a single text file.
PROB	IPCS	Enters a problem report in IPCS.
PROP	SFPROG	Provides Programmable Operator capability.
SAMGEN	INST	Loads and saves the CMSBAM saved segment.
SAVENC	INST	Reads a 3704/3705 Control Program load module into virtual storage, and extracts the control information required by CP. storage and save an image on a CP-owned disk.
SEGGEN	ADGCMS	Builds and saves a physical saved segment that is composed of one or more logical saved segments.
SNTMAP	PLNGDE	Processes the macro definitions in a system name table (SNT) file.
SPLOAD	INST	Loads files from the VM/SP product tape, source feature tape, or national language feature tape. and SFS directories during initial installation.
STAT	IPCS	Displays the status of reported system problems.
UTILITY	SERV	Performs installation and service utility tasks: obtains DASD information; creates stand-alone service programs on minidisk; writes an IPLable copy of the CP nucleus on tape; prints system definition files; creates a stand-alone service utility tape.
VMFAPPLY	SERV	Creates and/or updates auxiliary control files for the PTFs on the service tape.
VMFASM	SERV	Updates an ASSEMBLE source file according to entries in a control file, then assembles the updated file to produce an object file.
VMFBLD	SERV	Copies and renames PTF numbered text decks, applies patches, builds objects (nucleus, SFS load modules, RUNSTAF module).
VMFDOS	INST	Creates CMS files for VSE modules from VSE library distribution tape or SYSIN tape.
VMFHASM	SERV	Updates an ASSEMBLE source file according to entries in a control file, then assembles the updated file with the H-assembler to produce an object file.
VMFLKED	SERV	Link-edits modules into a load library (LOADLIB).

Table 5 (Page 4 of 4). CMS Commands Described in other Publications		
Command	Book Code	Usage
VMFLOAD	SERV	Punches a nucleus or stand-alone dump load deck.
VMFMAC	SERV	Builds macro libraries (MACLIBs) containing macro and copy files.
VMFMERGE	SERV	Applies PTFs to SNA products.
VMFNLS	SERV	Applies updates to national language files and compiles the updated versions.
VMFOVER	SERV	Creates a temporary product parameter file containing parameters for a single component and applies component parameter overrides.
VMFPLC2	SERV	Loads files from tape, dumps files to tape, and controls various tape drive operations.
VMFREC	SERV	Receives program update service or corrective service from tape.
VMFREMOV	SERV	Removes PTFs from SNA products.
VMFSETUP	SERV	Defines a minidisk and SFS directory access order.
VMFTXT	SERV	Builds a text library (TXTLIB) containing text decks.
VMFZAP	SERV	Applies ZAPs to SNA products.
VMSERV	SERV	Controls the individual service execs on the system Program Update Tape.
VRSIZE	SERV	Builds a DMKSLC TEXT file used to generate a virtual = rea (v = R) area when building the CP nucleus.
VSAMGEN	INST	Builds the CMSVSAM and CMSAMS physical saved segments.
VSAPL	OS PP	Uses VS APL interface in CMS.
VSEVSAM	INST	Loads VSE/VSAM assembler macros from tape and builds the VSEVSAM MACLIB.
ZAP	SERV	Modifies or dumps MODULE, LOADLIB, or TXTLIB files.
ZAPTEXT	SERV	Modifies or dumps individual text files.

Chapter 2. CMS Commands

This part contains reference information for the CMS commands used by general users. Each command description indicates the command format, operands and options; it also lists error messages and return codes the command issues. Usage notes are provided, where applicable.

For information about the System Product Interpreter, see the *VM/SP System Product Interpreter Reference*, and the *VM/SP System Product Interpreter User's Guide*.

For details on the XEDIT subcommands and macros, see *VM/SP System Product Editor Command and Macro Reference*. For usage information on XEDIT subcommands and macros, see *VM/SP System Product Editor User's Guide*.

For more detailed usage information on CMS commands, see the *VM/SP CMS User's Guide*.

The following commands and subcommands exist in the CP, CMS, and XEDIT environments:

CP QUERY SET

The following commands and subcommands exist in the CMS and XEDIT environments:

HELP LOAD SORT XEDIT

The following command and subcommand exists in the CP and XEDIT environments:

RESET

The command formats are presented in the following order:

- **Command Name:** Identifies the name of the command. The name is also included at the top of the page for easy reference.
- **Function Description:** Describes the general use of the command.
- **Format:** Lists the syntax (format) of the command with all the possible operands that you can use.
- **Operand and Option Description:** Describes the function of each operand and option and any values that you can include.
- **Usage Notes:** Contains notes about special uses of the command, its operands, or combinations of commands or operands.
- **Examples:** Provides one or more examples to show how the command is commonly used.
- **Responses:** Describes the responses sent to the terminal, caused by execution of the command. Some responses are command responses and are not to be construed as VM/SP system messages. When the command responses are not prefixed, they are not contained in *VM/SP System Messages and Codes*.
- **Messages and Return Codes:** Lists the messages issued by the command. In some cases, the text of a message is longer than a line on the display screen.

The message text may be divided in the middle of a word and continued on successive lines.

Refer to *VM/SP System Messages and Codes* for detailed information about the messages. Refer to *VM/SP System Messages Cross-Reference* for a listing of messages.

ACCESS

Use the ACCESS command to:

- identify a minidisk or Shared File System (SFS) directory to CMS,
- set up a list of the files on the specified minidisk or directory in your storage,
- establish a file mode letter for the files on a minidisk or in a directory.

Format

ACcess	<pre> [<i>dirid</i> <i>fm</i> [/<i>ext</i>] [(optionsA...[])]] [<i>vdev</i> <i>fm</i> [/<i>ext</i> [<i>fn</i> [<i>ft</i> [<i>fm</i>]]]]] [(optionsB...[])]] </pre> <p>OptionsA: [NOPROF] [NODISK]</p> <p>OptionsB: [NOPROF] [ERASE SAVEONLY NOSAVE] [NODISK]</p>
---------------	---

Operands

If you issue ACCESS without any operands, the 191 disk or top directory is accessed as file mode A.

dirid

identifies the SFS directory to be accessed. For this command you cannot use the *fm* format of *dirid*. For a more detailed description of *dirid*, refer to “Naming Shared File System (SFS) Directories” on page 4.

fm

assigns a one-character file mode letter to all files on the minidisk or directory being accessed.

ext

indicates the file mode of the parent minidisk or SFS directory. Files on the minidisk (*vdev*) or directory (*dirid*) being accessed are logically associated with files on the parent minidisk or directory. The minidisk or directory is considered a read-only extension. A parent minidisk or directory must be accessed in the search order before the extension. A blank must not precede or follow the slash.

vdev

makes available the minidisk at the specified virtual device address. Valid addresses are:

- 0001 through FFFF for a 370/XA mode virtual machine
- 001 through 5FF for a VM/SP virtual machine in basic control mode

- 001 through FFF for a System/370 mode virtual machine or VM/SP virtual machine in extended control mode.

On VM/SP and System/370 mode virtual machines you can supply a leading zero. For example,

```
access 0293 b
```

Note: In the preceding description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. Valid addresses in both environments are described to help you plan and develop applications that will run in both environments.

fn [ft [fm]]

defines a subset of the files on the specified minidisk. Only the specified files are included in the user file directory and only those files can be read. An asterisk coded in any of these fields indicates all file names, file types, or file mode numbers (except 0) are to be included. (See Usage Notes 2 and 3 under "Using the ACCESS command with *vdev*.") To specify a file mode, use a letter and a number, for example:

```
B1
```

For OS and DOS disk access restrictions, see Usage Note 8 under "Using the ACCESS command with *vdev*."

Options

NOPROF

suppresses execution of a PROFILE EXEC file. This option is valid only if the ACCESS command is the first command entered after you IPL CMS. Only the minidisk or directory at file mode A and the system disk (file mode S) with its associated extensions are accessed. On any ACCESS commands issued after you have IPLed CMS, the PROFILE EXEC is not executed and the NOPROF option is ignored.

NODISK

lets you gain access to the CMS operating system with no minidisks or directories accessed by CMS except the system disk (file mode S) and its extensions. This option is only valid if the ACCESS command is the first command you enter after you IPL CMS.

ERASE

indicates that you want to erase all of the files on the specified minidisk. This option is only valid for read/write disks. (See Usage Note 6 under "Using the ACCESS command with *vdev*").

SAVEONLY

accesses the minidisk if a saved copy of the file directory information is available in a saved segment. If it is not available, the minidisk is not accessed.

NOSAVE

accesses the minidisk and places the file directory information in your virtual machine. A saved copy of the file directory information in a saved segment is not used.

Usage Notes

Using the ACCESS command with either *dirid* or *vdev*

1. If you have a default file pool defined in your VM/SP directory, or if you define it when you IPL CMS, your top directory in that file pool is accessed as file mode A. Otherwise, if you have a defined disk address of 191 in your VM/SP directory, or you define it before you IPL CMS, your 191 disk is accessed as A. If you do not have either a top directory in the default file pool or a 191 disk defined, then nothing is accessed as file mode A.

Issuing ACCESS without any parameters or options accesses your top directory or 191 disk in the same manner.

2. If you have defined disk addresses 190, 192, and 19E in the VM/SP directory, or if they are defined before you IPL CMS, these disks are accessed as file mode S, D, and Y respectively. Following an IPL of CMS, you must issue explicit ACCESS commands to access other minidisks or directories. Ordinarily, you have access only to files with a file mode number of 2 on the system disk, or file mode S. Note that you cannot specify a file mode of S with the ACCESS command. When ACCESS is the first command issued after an IPL of the CMS system, a disk or SFS directory is not automatically defined as file mode A. Another ACCESS command must be issued to define a minidisk or directory as file mode A.

3. If you enter the ACCESS command and any associated operands or options at the VM READ after you IPL CMS, it must be entered in American English; you cannot use any other national language.

4. You can force a read/write minidisk or SFS directory into read-only status by accessing it as an extension of another minidisk, directory, or of itself; for example:

```
access 191 a/a
```

forces your 191 disk into read-only status.

Note: You can write to a file in a read-only SFS directory if you have write authority to the file.

5. When a minidisk or SFS directory is made a read-only extension of another minidisk or directory, some commands that allow you to specify a file mode will search extensions of the specified minidisk or directory. For a detailed description of read-only extensions, see the *CMS User's Guide*.
6. When you access a minidisk or SFS directory, a list of the files on the minidisk or directory is stored in your virtual machine. For shared minidisks, if another user updates a file on the minidisk, the minidisk's file directory is updated, but the list of files in your storage is not. You must reaccess the minidisk to get the current list of files.

For SFS directories, the list of files in your storage is automatically kept current for you. If another user updates a file in the SFS directory, your list of files is updated automatically and there is no need for you to reaccess the SFS directory.

7. If you are experiencing poor response time, and you have lots of directories or minidisks accessed, you should release the accessed directories and minidisks that you are not using. See the CMS RELEASE command.

Using the ACCESS command with *dirid*

1. When you access an SFS directory, the read/write status is determined by who owns it.
 - If you are the directory owner, then it is accessed as read/write.
 - If you are not the directory owner, then the directory is accessed as read-only.
 - If you attempt to access a directory for which you are not authorized, the directory is not accessed.
2. Reading and writing to another user's files in the Shared File System is controlled at the file level. Therefore, although you can only access another user's directory as read-only, you can write to files for which you have been granted write authority. Refer to the *VM/SP CMS User's Guide* for more detail.
3. If a directory is locked EXCLUSIVE, only the holder of the lock can access the directory. If a directory is locked SHARE or UPDATE, any authorized user can access the directory.
4. You can access a directory as read-only even if it has no files in it.

Using the ACCESS command with *vdev*

1. Associated with each CMS minidisk is a file directory, which contains an entry for every CMS file on the minidisk. Specifying ACCESS without the SAVEONLY or NCSAVE options accesses the minidisk using a saved copy of the file directory that contains entries for only those files that you can reference. If the saved segment containing the saved copy is not available or is not current, ACCESS creates a file directory in your virtual machine.

If you use the CP LINK command to link to a new minidisk, issue an ACCESS command each time. Do this so that you obtain the appropriate file directory.

2. The *fn ft fm* fields can only be specified for minidisks that are accessed as read-only extensions. Also, requesting a subset of files creates a file directory in your virtual machine. For example:

```
access 195 b/a * assemble
```

gives you read-only access to all the files with a file type of ASSEMBLE on the minidisk at virtual address 195, and the file directory information is stored in your virtual machine. The command:

```
access 190 z/a * * z1
```

gives you access to all files on the system disk (190) that have a file mode number of 1.

When you access any minidisk in read-only status, files with a file mode number of 0 are not accessed.

3. You can also identify a set of files on a minidisk by referring to a file name or file type prefix. For example:

```
access 192 c/a abc*
```

accesses only those files in the minidisk at virtual address 192 whose file names begin with the characters ABC. The command:

```
access 192 c/a * a* c2
```

gives you access to all files whose file types begin with an A and that have a file mode number of 2.

4. Accessing the same minidisk with different file modes affects the type of access. Once a minidisk is accessed using a saved file directory, subsequent ACCESS commands for the same minidisk place the file directory in your virtual machine. For example, the command sequence

```
access 19f g
access 19f h
access 19f i
```

accesses the minidisk defined at address 19F with a saved file directory for the file mode G minidisk while the minidisks with file mode H and I are accessed with the file directory in your virtual machine.

5. When you have linked to a formatted minidisk as read-only, and it does not contain any files, you cannot access the minidisk. A formatted minidisk linked as read/write can be accessed whether or not it contains files.
6. If you enter the ERASE option by mistake, you can recover from the error as long as you have not yet written any new files onto the minidisk. (That is, you have not yet caused CMS to rewrite the file directory.) Reissue the ACCESS command without the ERASE option.
7. You should never attempt to access a minidisk in read/write status if another user already has it in read/write status; the results are unpredictable.
8. When accessing OS and DOS disks:
 - a. You cannot specify file name, file type and file mode when you access OS or DOS disks, nor can you specify any options.
 - b. In order to see OS and DOS disks, you must have a read/write minidisk or SFS directory available as file mode A, if you are going to use the LOAD command with the MAP option. (MAP is a default option.)
 - c. If two or more minidisks have been accessed in CMS, and CP DEFINE commands are executed that swap virtual addresses, then a subsequent RELEASE command may write the file directory on the wrong minidisk; for example:

```
(CMS) ACCESS 193 C
(CMS) ACCESS 198 E
(CP)  DEFINE 193 293
(CP)  DEFINE 198 193
(CMS) RELEASE C
```

This sequence of commands will write the file directory from 193 to 198 since the CP definitions are unknown to CMS.

Example

Assuming that a default file pool has been set, to access an SFS directory called .PROJ1 as your file mode A, enter:

```
access .proj1 a
```

To access a minidisk defined at address 498 as file mode B, enter the following:

```
access 498 b
```

To access a minidisk defined at address 498 as file mode B (only if a saved copy of the directory is available), enter the following:

```
access 498 b (saveonly)
```

Responses

DMSACP723I *mode (vdev|dirname) {R/O|R/W} [-OS|-DOS]*

This message is displayed if the specified minidisk or directory is a read-only CMS minidisk or read-only SFS directory. If the minidisk is in OS or DOS format, the message indicates the format, as well as whether it is a read/write or read-only minidisk.

DMSACC724I *vdev1|dirname1 replaces mode (vdev2|dirname2)*

Before execution of the command, the minidisk represented by *vdev2*, or the directory represented by *dirname2*, was accessed as *mode*. The minidisk *vdev1*, or directory *dirname1* is now assigned that file mode letter.

DMSACP725I *vdev|dirname also = mode [-OS|-DOS] minidisk*

The minidisk specified by *vdev*, or directory specified by *dirname*, is accessed as *mode* and an ACCESS command was issued to assign it another file mode letter.

DMSACP726I *vdev|dirname mode released*

The minidisk being accessed at virtual address *vdev* as a read/write minidisk, or the directory *dirname*, is already accessed at a different mode. It is released from that mode.

Messages and Return Codes

DMSACC017E Invalid device address *vdev* [RC=24]
 DMSACR048E Invalid filemode *fm* [RC=24]
 DMSACC059E *vdev|dirname* already accessed as read/write filemode *fm* [RC=36]
 DMSACP060E File(s) *fn [ft [fm]]* not found; disk *fm(vdev)* will not be accessed [RC=28]
 DMSACC066E *option1* and *option2* are conflicting options [RC=24]
 DMSACC109S Virtual storage capacity exceeded [RC=104]
 DMSACP112S Disk *fm(vdev)* device error [RC=100]
 DMSACP113S *fm(vdev)* not attached [RC=100]
 DMSACP230W O/S disk--fileid and/or options specified are ignored [RC=4]
 DMSACP1078E Cannot access saved file directory for this disk [RC=44]
 DMSACR1184E Directory *dirname* not found or you are not authorized for it [RC=100]
 DMSACR1210E Directory *dirname* not found [RC=100]
 DMSACR1216E Option *option* is not valid when used for a directory [RC=24]
 DMSACR1216E Parameter *parameter* is not valid when used for a file in a directory [RC=24]
 DMSJED1223E There is no default file pool currently defined [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

AMSERV

Use the AMSERV command to invoke access method services to:

- Define VSAM catalogs, data spaces, or clusters
- Alter, list, copy, delete, export or import VSAM catalogs and data sets.

Format

AMserv	$fn1 \left[\begin{array}{l} fn2 \\ \underline{fn1} \end{array} \right] \left[(\text{options... } []) \right]$ <p>Options: [PRINT] [TAPIN $\left\{ \begin{array}{l} 18n \\ \text{TAP}n \end{array} \right\} \left. \right]$ [TAPOUT $\left\{ \begin{array}{l} 18n \\ \text{TAP}n \end{array} \right\} \left. \right]$</p>
---------------	---

Operands

fn1

specifies the file name of a CMS file with a file type of AMSERV that contains the access method services control statements to be executed. CMS searches all of your accessed disks and SFS directories, using the standard search order, to locate the file.

fn2

specifies the file name of the CMS file that is to contain the access method services listing; the file type is always LISTING. If *fn2* is not specified, the LISTING file will have the same name as the AMSERV input file (*fn1*).

The LISTING file is written to the first read/write disk or directory in the standard search order, usually your disk or directory accessed as A. If a LISTING file with the same name already exists, it is replaced.

Options

PRINT

spools the output listing to the virtual printer, instead of writing it to a disk or directory. If PRINT is specified, *fn2* cannot be specified.

TAPIN 18*n*

TAPIN TAP*n*

specifies that tape input is on the tape drive at the address indicated by 18*n* or TAP*n*. The number, *n*, may be 1, 2, 3, or 4, indicating virtual addresses 181 through 184, respectively.

TAPOUT 18*n*

TAPOUT TAP*n*

specifies that tape output should be written to the tape drive at the address indicated by 18*n* or TAP*n*. The number, *n*, may be 1, 2, 3, or 4, indicating virtual addresses 181 through 184, respectively.

Note: If both TAPIN and TAPOUT are specified, their virtual device addresses must be different.

Usage Notes

1. To create a job stream for access method services, you can use an editor to create a file with the file type of AMSERV. The editor automatically sets input margins at columns 2 and 72.
2. Restrictions placed on VSAM usage in CMS are listed in this publication in "Appendix A: VSE/VSAM Functions Not Supported in CMS" and "Appendix B: OS/VS Access Method Services and VSAM Functions Not Supported in CMS." Refer to *Using VSE/VSAM Commands and Macros* for a description of access method services control statements format and syntax.
3. You must use the DLBL command to identify the master catalog. Input and output files may also require a DLBL command. For more information on DLBL requirements for AMSERV see *VSE/VSAM Programmer's Reference*.
4. When you use tape input and/or output with the AMSERV command, you are prompted to enter the ddnames; a maximum of 16 ddnames are allowed for either input or output. The ddnames can each have a maximum of seven characters and must be separated by blanks.

While using AMSERV, only one tape at a time can be attached for either input or output. If you enter more than one tape ddname, specify the tape files in the sequence they are used in the input stream.

5. A CMS format variable file cannot be used directly as input to AMSERV functions as a variable (V) or variable blocked (VB) file because the standard variable CMS record does not contain the BL and RL headers needed by the variable record modules. If these headers are not included in the record, errors will result.

Most files written by AMSERV will show a RECFM of V, even if the true format is fixed (F), fixed blocked (FB), undefined (U), variable or variable blocked. The programmer must know the true format of the file he is trying to use with the AMSERV command and access it properly, or errors will result.
6. If an AMSERV command abnormally terminates or you issue HX to terminate an AMSERV command, the AMSERV environment may not be reset correctly. If a subsequent AMSERV abends, you must re-IPL CMS.
7. If the AMSERV input file contains the access method services control statement "DELETE" with "IGNORERROR," the PRINT option on the AMSERV command must be used to send the listing to the virtual printer.
8. The density used for tapes is the default density of the tape drive or the density of the tape mounted on it.

Additional Note for CMS/DOS Users:

AMSERV internally issues an ASSGN command for SYSIPT and locates the source file; therefore, you do not need to assign it. If you use the TAPIN or TAPOUT options, AMSERV also issues ASSGN commands for the tape drives (assigning logical units SYS004 and SYS005).

Any other assignments and DLBL definitions that are in effect when you invoke the AMSERV command are saved and restored when the command completes executing.

Example

If you enter the AMSERV command with no options, for example:

```
amserv myfile
```

you will get a CMS file with a file name of MYFILE and a file type of LISTING. LISTING files take up considerable space, so you should erase them as you no longer need them. If you do not want to create a file from the listing, you can have the output spooled to the virtual printer. The following command:

```
amserv myfile (print
```

spools the output to the virtual printer and no file is created.

Responses

The CMS ready message indicates that access method services has completed processing. If access method services completed with a nonzero return code, the return code is shown in the ready message. Examine the LISTING file created by AMSERV to determine the results of access method services processing.

The publication *VSE/VSAM Messages and Codes* lists and explains the messages access method services generates and the associated reason codes.

DMSAMS367R Enter tape {input|output} DDNAMEs:

This message prompts you to enter the ddnames associated with the tape files.

DMSAMS722I File *fn* LISTING *fm* will hold AMSERV output

This message is displayed when you enter a *fn2* operand or when the listing is not being written on your disk or directory accessed as A; it tells you the file identifier of the output listing.

Messages and Return Codes

DMSAMS001E	No filename specified [RC=24]
DMSAMS002E	File FNAME1 AMSERV not found [RC=28]
DMSAMS003E	Invalid option: <i>option</i> [RC=24]
DMSAMS006E	No read/write filemode accessed for FNAME2 LISTING [RC=36]
DMSAMS007E	File FNAME1 AMSERV <i>fm</i> not fixed, 80-character records [RC=32]
DMSAMS065E	<i>option</i> option specified twice [RC=24]
DMSAMS066E	<i>option1</i> and <i>option2</i> are conflicting options [RC=24]
DMSAMS070E	Invalid parameter <i>parameter</i> [RC=24]
DMSAMS109S	Virtual storage capacity exceeded [RC=104]
DMSASN113S	Tapn(<i>vdev</i>) not attached [RC=100]
DMSAMS136S	Unable to load IDCMA5 [RC=104]
DMSAMS228E	No DDNAME entered [RC=24]

ASSEMBLE

Use the ASSEMBLE command to invoke the assembler to assemble a file containing source statements. Assembler processing and output is controlled by the options selected.

Format

Assemble	<p><i>fn</i> [(options... [])]</p> <p>Listing Control Options:</p> <p>[<u>ALOGIC</u>] [<u>ESD</u>] [<u>FLAG</u> (<i>nnn</i>)] [<u>LINECOUN</u> (<i>nn</i>)] [<u>NOALOGIC</u>] [<u>NOESD</u>] [<u>FLAG</u> (0)] [<u>LINECOUN</u> (55)]</p> <p>[<u>LIST</u>] [<u>MCALL</u>] [<u>MLOGIC</u>] [<u>RLD</u>] [<u>LIBMAC</u>] [<u>NOLIST</u>] [<u>NOMCALL</u>] [<u>NOMLOGIC</u>] [<u>NORLD</u>] [<u>NOLIBMAC</u>]</p> <p>[<u>XREF</u> (FULL)] [<u>PRINT</u>] [<u>XREF</u> (SHORT)] [<u>NOPRINT</u>] [<u>NOXREF</u>] [<u>DISK</u>]</p> <p>Output Control Options:</p> <p>[<u>DECK</u>] [<u>OBJECT</u>] [<u>TEST</u>] [<u>NODECK</u>] [<u>NOOBJECT</u>] [<u>NOTEST</u>]</p> <p>SYSTEM Options:</p> <p>[<u>NUMBER</u>] [<u>STMT</u>] [<u>TERMINAL</u>] [<u>NONUM</u>] [<u>NOSTMT</u>] [<u>NOTERM</u>]</p> <p>Other Assembler Options:</p> <p>[<u>ALIGN</u>] [<u>BUFSIZE</u> (MIN)] [<u>RENT</u>] [<u>NOALIGN</u>] [<u>BUFSIZE</u> (STD)] [<u>NORENT</u>] [<u>BUFSIZE</u> (MAX)]</p> <p>[<u>YFLAG</u>] [<u>SYSPARM</u> (<i>string</i>)] [<u>WORKSIZE</u> (2048K)] [<u>NOYFLAG</u>] [<u>SYSPARM</u> ()] [<u>WORKSIZE</u> (<i>nnnnnK</i>)] [<u>SYSPARM</u> (?)]</p>
----------	--

Operands

fn

is the file name of the source file to be assembled and/or the file name of assembler output files. The file must have fixed-length, 80-character records. By default, the assembler expects a CMS file with a file type of ASSEMBLE.

Listing Control Options

The list below describes the assembler options you can use to control the assembler listing.

ALOGIC

lists conditional assembly statements in open code.

NOALOGIC

suppresses the ALOGIC option.

ESD

lists the external symbol dictionary (ESD).

NOESD

suppresses the printing of the ESD listing.

FLAG (*nnn*)

FLAG (0)

does not include diagnostic messages and MNOTE messages below severity code *nnn* in the listing. Diagnostic messages can have severity codes of 4, 8, 12, 16, or 20 (20 is the most severe); and MNOTE message severity codes can be between 0 and 255. For example, FLAG (8) suppresses diagnostic messages with a severity code of 4 and MNOTE messages with severity codes of 0 through 7.

LINECOUN (*nn*)

LINECOUN (55)

nn specifies the number of lines to be listed per page.

LIST

produces an assembler listing. Any previous listing is erased.

NOLIST

does not produce an assembler listing. However, any previous listing is still erased. This option overrides ESD, RLD, and XREF.

MCALL

lists the inner macro instructions encountered during macro generation following their respective outer macro instructions. The assembler assigns statement numbers to these instructions. The MCALL option is implied by the MLOGIC option; NOMCALL has no effect if MLOGIC is specified.

NOMCALL

suppresses the MCALL option.

MLOGIC

lists all statements of a macro definition processed during macro generation after the macro instruction. The assembler assigns statement numbers to them.

NOMLOGIC

suppresses the MLOGIC option.

RLD

produces the relocation dictionary (RLD) as part of the listing.

NORLD

does not print the relocation dictionary.

LIBMAC

lists the macro definitions read from the macro libraries and any assembler statements following the logical END statement. The logical END statement is the first END statement processed during macro generation. It may appear in a macro or in open code; it may even be created by substitution. The assembler

assigns statement numbers to the statements that follow the logical **END** statement.

NOLIBMAC

suppresses the **LIBMAC** option.

XREF (FULL)

includes in the assembler listing a cross-reference table of all symbols used in the assembly. This includes symbols that are defined but never referenced. The assembler listing also contains a cross-reference table of literals used in the assembly.

XREF (SHORT)

includes in the assembler listing a cross-reference table of all symbols that are referenced in the assembly. Any symbols defined but not referenced are not included in the table. The assembler listing contains a cross-reference table of literals used in the assembly.

NOXREF

does not print the cross-reference tables.

PRINT**PR**

writes the **LISTING** file to the printer.

NOPRINT**NOPR**

suppresses the printing of the **LISTING** file.

DISK**DI**

places the **LISTING** file on a disk or **SFS** directory, searching for the disk or directory in the following order:

1. An accessed read/write disk or directory from which the assemble source is read.
2. The read/write "parent" disk or directory, if the assemble source is read from a disk or directory that is accessed as a read-only extension.
3. The disk or directory accessed as **A** with read/write access.

If the disk or directory accessed as **A** is not accessed **R/W** and the first two conditions do not apply, then an error message is generated.

Output Control Options

The output control options are used to control the object module output of the assembler.

DECK

writes the object module on the device specified on the **FILEDEF** statement for **PUNCH**. If this option is specified with the **OBJECT** option, the object module is written both on the **PUNCH** and **TEXT** files.

NODECK

suppresses the **DECK** option.

OBJECT**OBJ**

writes the object module on the device, which is specified by the FILEDEF statement for TEXT, and erases any previous object modules. If this option is specified with the DECK option, the object module is written on the two devices specified in the FILEDEF statement for TEXT and PUNCH.

NOOBJECT**NOOBJ**

does not create the object module. However, any previous object module is still erased.

TEST

includes the special source symbol table (SYM cards) in the object module. This option should not be used for programs to be run under CMS because the SYM cards are not acceptable to the CMS LOAD and INCLUDE commands.

NOTEST

does not produce SYM cards.

SYSTEM Options

The SYSTEM options are used to control the SYSTEM file associated with your assembly.

NUMBER**NUM**

writes the line number field (columns 73-80 of the input records) in the SYSTEM listing for statements for which diagnostic information is given. This option is valid only if TERMINAL is specified.

NONUM

suppresses the NUMBER option.

STMT

writes the statement number assigned by the assembler in the SYSTEM listing for statements for which diagnostic information is given. This option is valid only if TERMINAL is specified.

NOSTMT

suppresses the STMT option.

TERMINAL**TERM**

writes the diagnostic information on the SYSTEM data set. The diagnostic information consists of the diagnosed statement followed by the error message issued.

NOTERM

suppresses the TERMINAL option.

Other Assembler Options

The following options allow you to specify various functions and values for the assembler.

ALIGN**ALGN**

aligns all data on the proper boundary in the object module; for example, an F-type constant is aligned on a fullword boundary. In addition, the assembler checks storage addresses used in machine instructions for alignment violations.

NOALIGN**NOALGN**

does not align data areas other than those specified in CCW instructions. The assembler does not skip bytes to align constants on proper boundaries. Alignment violations in machine instructions are not diagnosed.

BUFSIZE (MIN)

uses the minimum buffer sizes (790 bytes) for each of the utility data sets (SYSUT1, SYSUT2, and SYSUT3). Storage normally used for buffers is allocated to work space. Because more work space is available, more complex programs can be assembled in a given virtual storage size; but the speed of the assembly is substantially reduced.

BUFSIZE (STD)

chooses the buffer size that gives optimum performance. The buffer size depends on the amount of virtual storage. Of the assembler working storage in excess of minimum requirements, 37% is allocated to the utility data set buffers and the rest to macro generation dictionaries.

BUFSIZE (MAX)

is useful when many macros and/or large macros are used in an assembly. The assembler uses up to 15 save areas for input records and saves the areas according to their frequency of use, optimizing the macro generation phase. This option has no effect unless a large enough region is available. The number of allocated save areas is printed on the statistics page of the assembler listing. Refer to the *OS/VS-VM/370 Assembler Programmer's Guide*, Appendix E for a description of the effects of BUFSIZE.

RENT

checks your program for a possible violation of program re-enterability. Code that makes your program nonre-enterable is identified by an error message.

NORENT

suppresses the RENT option.

YFLAG

does not suppress the warning messages that indicate that relocatable Y-type address constants have been declared.

NOYFLAG

suppresses the warning messages that indicate relocatable Y-type constants have been declared.

SYSPARM (string)**SYSPARM ()****SYSPARM (?)**

passes a character value to the system variable symbol, SYSPARM. The variable (string) may be up to 100 characters long, and may not contain any blanks or parentheses. If you want to enter a string containing blanks or parentheses, use the SYSPARM (?) format. With the SYSPARM (?) format, CMS prompts you with the message:

ENTER SYSPARM:

You can enter up to 100 characters. SYSPARM () enters a null string of characters.

Note: If ASSEMBLE is called as a command, the SYSPARM information is translated to uppercase.

WORKSIZE (2048K)

WORKSIZE (nnnnnK)

allows the user to delimit the use of region space. The specified value does not include the space for modules and system areas. The allowed range is from 32K to 10240K. The virtual machine size must be large enough to accommodate the WORKSIZE option; otherwise the option has no effect.

Usage Notes

1. When you issue the ASSEMBLE command, default FILEDEF commands are issued for assembler data sets. You may want to override these with explicit FILEDEF commands. The ddnames used by the assembler are:

ASSEMBLE (SYSIN input to the assembler)

TEXT (SYSLIN output of the assembler)

LISTING (SYSPRINT output of the assembler)

PUNCH (SYSPUNCH output of the assembler)

CMSLIB (SYSLIB input to the assembler)

SYSUT1 (workfile of the assembler)

SYSUT2 (workfile of the assembler)

SYSUT3 (workfile of the assembler)

The default FILEDEF commands issued by the assembler for these ddnames are:

```
FILEDEF ASSEMBLE DISK fn ASSEMBLE fm (RECFM FB LRECL 80 BLOCK 800
```

```
FILEDEF TEXT DISK fn TEXT fm
```

```
FILEDEF LISTING DISK fn LISTING fm (RECFM FBA BLOCK 1210
```

```
FILEDEF PUNCH PUNCH
```

```
FILEDEF CMSLIB DISK CMSLIB MACLIB * (RECFM FB LRECL 80 BLOCK 800
```

```
FILEDEF SYSUT1 DISK fn SYSUT1 fm4 (BLOCK 7294 AUXPROC asmproc
```

```
FILEDEF SYSUT2 DISK fn SYSUT2 fm4 (BLOCK 7294 AUXPROC asmproc
```

```
FILEDEF SYSUT3 DISK fn SYSUT3 fm4 (BLOCK 7294 AUXPROC asmproc
```

At the completion of the ASSEMBLE command, all FILEDEFs that do not have the PERM option are erased.

2. If you want to use any CMS macro or copy libraries during an assembly, issue the GLOBAL command to identify the macro libraries before you issue the ASSEMBLE command. For example:

```
global maclib dmssp cmslib osmacro testlib
```

identifies the MACLIB files named CMSLIB, DMSSP, OSMACRO, and TESTLIB.

3. To use OS macro libraries during an assembly, issue the FILEDEF command for the OS data set. Use a ddname of CMSLIB and assign a CMS file identifier; the file type must be MACLIB, and you must use the file name on the GLOBAL command line. For example:

```
filedef cmslib disk oldtest maclib c dsn oldtest macros
```

```
global maclib oldtest
```

ASSEMBLE

assigns the OS data set OLDTEST.MACROS, on the disk or directory accessed as C, a CMS file ID of OLDTEST MACLIB and identifies it as the macro library to be used during assembly.

4. You cannot assemble programs using DOS macros from the DOS/VS source statement libraries under CMS/DOS. You should use the SSERV, ESERV, and MACLIB commands to create CMS MACLIBs to contain DOS macros for assembly under CMS/DOS. See *VM/SP Application Development Guide for CMS* for examples.
5. You need not make any logical assignments for input or output files when you use the assembler under CMS/DOS. File definitions are assigned by default under CMS, as described in Usage Note 1.
6. ASSEMBLE uses the extended plist for processing the SYSPARM parameter. If you are calling ASSEMBLE from an assembler language program and using SYSPARM, you should supply an extended plist. *VM/SP Application Development Guide for CMS* has more information on how an assembler language program can supply an extended plist.
7. Usage information about the VM/SP Assembler Language and assembler options can be found in *OS/VS and VM/370 Assembler Programmer's Guide* and *OS/VS, DOS/VS, and VM/370 Assembler Language*.
8. When assembling large files, you may want to issue the FILEDEF command for the SYSUT n data sets to avoid a filling a disk or your file space.

When you issue the ASSEMBLE command, CMS builds the SYSUT1 data set in storage until there is no more space, and then puts the file on a disk or directory. The SYSUT1 file has a record length of 8000 and a fixed record format. To avoid filling the disk or your file space prematurely, issue the FILEDEF command for the SYSUT n data sets, and they will not be built in storage. Instead, the SYSUT n files are built only on disk or in a directory with a record length of 7294 and a variable record format, which uses less space.

Example

Specifying the following:

```
assemble myfile (print
```

assembles the assembler language source file MYFILE ASSEMBLE into object module format and directs the output listing to printer.

Messages and Return Codes

For the messages and return codes associated with the ASSEMBLE command, see the *OS/VS and VM/370 Assembler Programmer's Guide*.

ASSGN

Use the ASSGN command in CMS/DOS to assign or unassign a system or programmer logical unit for a virtual I/O device.

Format

ASSGN	SYS <i>xxx</i> { <ul style="list-style-type: none"> Reader PUnch PRinter Terminal } { <ul style="list-style-type: none"> TAP [<i>n</i>] [<u>1</u>] } <i>mode</i> <ul style="list-style-type: none"> IGN UA 	[(options... [])]
	Options: [<u>UPCASE</u>] [7TRACK] [TRTCH <i>a</i>] [DEN <i>den</i>] [<u>LOWCASE</u>] [9TRACK]	

Operands

SYS*xxx*

specifies the system or programmer logical unit to be assigned to a particular physical device. SYS000 through SYS241 are valid programmer logical units in CMS/DOS; they may be assigned to any valid device. The system logical units you may assign, and the devices to which they may be assigned, are:

SYS*xxx* Valid Assignments

SYSRDR	Reader, disk or directory,tape
SYSIPT	Reader, disk or directory,tape
SYSIN	Reader, disk or directory,tape
SYSPCH	Punch, disk or directory,tape
SYSLST	Printer, disk or directory,tape
SYSLOG	Terminal, printer
SYSOUT	Tape
SYSSLB	Disk or directory
SYSRLB	Disk or directory
SYSCLB	Disk or directory
SYSCAT	Disk or directory

The assignment of a system logical unit to a particular device type must be consistent with the device type definition for the file in your program.

Reader

is the spooled card reader (card reader I/O must not be blocked).

PUnch

is the spooled punch.

PRinter
is the spooled printer.

Terminal
is your terminal (terminal I/O must not be blocked).

TAP[n]
is a magnetic tape. n is the symbolic number of the tape drive. It is either 1, 2, 3, or 4, representing virtual addresses 181, 182, 183, and 184, respectively. If n is omitted, TAP1 is assumed.

mode
specifies the one-character mode letter of the disk or directory being assigned to the logical unit (SYSxxx). The disk or directory must be accessed when the ASSGN command is issued. SYSRDR, SYSIPT, and SYSIN cannot be assigned to a DOS-formatted FB-512 disk.

IGN
means that any attempt to read from the specified device results in an end-of-file indication, and that any attempt to write to the device is ignored. IGN is not valid when associated with SYSRDR, SYSIPT, SYSIN, or SYSCLB.

UA
indicates that the logical unit is to be unassigned. When you release a disk or directory for which an assignment is active, it is automatically unassigned.

Options

UPCASE
translates all terminal input data to uppercase.

LOWCASE
retains all terminal input data as keyed in.

7TRACK

9TRACK
is the tape setting.

TRTCH a
refers to the tape recording technique for 7-track tapes. Use the following chart to determine the value of a.

a	Parity	Converter	Translator
O	odd	off	off
OC	odd	on	off
OT	odd	off	on
E	even	off	off
ET	even	off	on

DEN den
is tape density: den can be 200, 556, 800, 1600, or 6250 bits per inch (bpi). If 200 or 556 are specified, 7TRACK is assumed. If 800, 1600, or 6250 are specified, 9TRACK is assumed. (See Usage Note 8.)

Usage Notes

1. When you enter the CMS/DOS environment with the command SET DOS ON, SYSLOG is assigned by default to TERMINAL. If you specify the mode letter of the VSE system residence on the SET DOS ON command line, SYSRES is assigned to that mode.
2. You cannot assign any of the following VSE system logical units with the ASSGN command:


```
SYSRES  SYSLNK  SYSDMP
SYSCTL  SYSREC
```
3. If you assign the logical unit SYSIN to a virtual device, SYSRDR and SYSIPT are also assigned to that device. If you make a logical assignment for SYSOUT, both SYSLST and SYSPCH are assigned.
4. To obtain a list of current assignments, use the LISTIO command.
5. To cancel all current assignments (that is, to unassign them), you can enter, in succession, the commands:


```
set dos off
set dos on [mode]
```
6. If you want to access VSE private libraries, you must assign the logical units SYSSLB (source statement library), SYSRLB (relocatable library), and SYSLCB (core image library), and you must issue the DLBL command to establish a file definition.
7. An assignment to *mode* should be accompanied by a DLBL command that provides the file identification.
8. If no tape options are specified, the default for a 7-track tape is 800 bpi, data converter off, translator off, and odd parity. If the tape is 9-track, the density defaults to the highest density of the tape drive. 1600 bpi is the default for 9-track 800/1600 dual-density tapes and 6250 bpi is the default for 9-track 1600/6250 dual-density tapes. If the tape drive is phase-encoded, density defaults to the density of the tape. If the tape drive is NRZI, the reset condition is 800 bpi.
9. 8809 tape drives require the 9TRACK and DEN 1600 options. These are the default options; it is not necessary to state them explicitly.
10. Assignment of Programmer Logical units to 'T' and 'R' is restricted to terminal and reader respectively.

Example

To assign logical unit SYS010 to a the disk or directory accessed as B, enter the following:

```
assgn sys010 b
```

Messages and Return Codes

DMSASN003E	Invalid option: <i>option</i> [RC=24]
DMSASN027E	Invalid device <i>devtype</i> [for SYSaaa] [RC=24]
DMSASN028E	No logical unit specified [RC=24]
DMSASN029E	Invalid parameter <i>parameter</i> in the option <i>option</i> field [RC=24]
DMSASN035E	Invalid tape mode [RC=24]
DMSASN050E	Parameter missing after SYSaaa [RC=24]
DMSASN065E	<i>option</i> option specified twice [RC=24]
DMSASN066E	<i>option1</i> and <i>option2</i> are conflicting options [RC=24]
DMSASN069E	Filemode <i>mode</i> not accessed [RC=36]

ASSGN

DMSASN070E Invalid parameter *parameter* [RC = 24]
DMSASN087E Invalid assignment of *SYSaaa* to device *devtype* [RC = 24]
DMSASN090E Invalid device class *devclass* for *devtype* [RC = 36]
DMSASN099E CMS/DOS environment not active [RC = 40]
DMSASN113S *Tapn(vdev)* not attached [RC = 100]

CATCHECK

As a CMS VSAM user (with or without DOS set ON), you can use the CATCHECK command to invoke the VSE/VSAM Catalog Check Service Aid to verify a complete catalog structure. CATCHECK produces a print file containing the catalog analysis. If you do not specify a catalog name with CATCHECK, it uses the default catalog specified via the DLBL command.

Format

CATCHECK	[<i>catname</i> <i>catname/password</i>]
-----------------	---

Operands

catname

is the catalog name of the catalog to be checked. The name can be a maximum of 44 characters and must follow the VSE/VSAM catalog naming conventions.

password

is the password for the catalog "catname" as specified when the catalog was defined. The maximum length is 8 characters. If you specify the password, you must put a slash between the catalog name and the password.

Usage Notes

1. If a catalog name is not specified on the command line, the default catalog is used. The default catalog is the job catalog identified via a ddname of "IJSYSUC" on the DLBL command. If a job catalog was not specified, the default catalog name will be the master catalog identified via the ddname of "IJSYSCT" on the DLBL command.
2. When a catalog name is specified, a DLBL need not be issued for the catalog if it is not the master catalog. A DLBL for the master catalog must always be in effect when running VSAM.
3. The output must always go to the virtual printer.
4. CATCHECK uses the extended plist for processing the *catname/password* parameter. If you are calling CATCHECK from an assembler language program and using *catname* or *catname/password*, you should supply an extended plist. *VM/SP Application Development Guide for CMS* has more information on how an assembler language program can supply an extended plist.

Example

If you have only issued a DLBL command for the master catalog, then specifying the following:

```
catcheck private.cat1
```

produces a print file containing the catalog analysis for PRIVATE.CAT1 catalog.

CATCHECK

Messages and Return Codes

DMSCCK109S	Virtual storage capacity exceeded [RC = 104]
DMSCCK803E	Invalid parameter specification [RC = 4]
DMSCCK804S	Error establishing CMS/DOS environment [RC = 8]
DMSCCK805S	Error assigning output to printer [RC = 12]
DMSCCK806S	VSE/VSAM phase IKQVCHK not found [RC = 16]
DMSCCK807S	Error encountered issuing ASSGN for catalog [RC = 20]

CMDCALL

Use the CMDCALL command to convert EXEC 2 extended PLIST function calls to CMS extended or standard PLIST command calls.

Format

CMDCALL	<i>[cmd [operand1 [operand2 ... operandn]]]</i>
----------------	---

Operands

cmd

is the command that is to be invoked with the CMS extended PLIST, indicating invocation as a command, rather than as a function.

operand1 operand2 ...

are the operands to be passed with the command.

Usage Notes

1. The extended PLIST and the standard CMS PLIST are adjusted for the command or function called, and that command or function is invoked via SVC 204.
2. If an extended PLIST is not available, the command or function called is invoked only with the standard PLIST adjusted for the command or function called.
3. CMDCALL invoked with no calling command or function returns a return code of zero. Otherwise, the return code is that of the command invoked via the CMDCALL function.

Example

For example, if, from within your EXEC 2 exec, you want to erase the file TEST EXEC from your disk or directory accessed as A, you would specify the following:

```
cmdcall erase test exec a
```

CMSBATCH

Use the CMSBATCH command to invoke the CMS batch facility. Instead of compiling or executing a program interactively, virtual machine users can transfer jobs to the virtual card reader of an active CMS batch virtual machine. This frees their terminals for other work.

Format

CMSBATCH	[<i>sysname</i>]
-----------------	--------------------

Operands

sysname

is the eight-character identification of the saved system that is specifically generated for CMS batch operations via the CP SAVESYS command and the NAMESYS macro. Refer to the *VM/SP Application Development Guide for CMS* publication for details on SAVESYS and NAMESYS use.

Note: If *sysname* is not supplied on the command line, then the system that the system operator is currently logged onto becomes the CMS batch virtual machine.

Usage Notes

1. The CMSBATCH command must be invoked immediately after an IPL of the CMS system. Alternatively, BATCH may be specified following the PARM operand on the IPL command line.
2. Do not issue the CMSBATCH command if you use a virtual disk at address 195; the CMS batch virtual machine erases all files on the disk at address 195.
3. For a description of how to send jobs to the CMS batch virtual machine, see the *VM/SP CMS User's Guide*. For an explanation of setting up a batch virtual machine, see the *VM/SP Operator's Guide*.
4. The CMS batch virtual machine can be utilized by personnel who do not have access to a terminal or a virtual machine. This is accomplished by submitting jobs via the real card reader. For details on this, see the *VM/SP CMS User's Guide*.
5. If the CMSBATCH command encounters recursive abends, the message "CMSBATCH system ABEND" appears on the system operator's console.

Messages and Return Codes

DMSBTB100E	No batch processor available [RC = 40]
DMSBTB101E	Batch not loaded [RC = 31 55 70 76 88 99]
DMSBTP105E	No job card provided
DMSBTP106E	/JOB card format invalid
DMSBTP107E	CP/CMS command <i>command</i> [, <i>devtype</i>] not allowed [RC = 100]
DMSBTP108E	/SET card format invalid [RC = 88]
DMSBTP109E	{CPU Printer Punch} limit exceeded

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in loading a file	322

CMSSERV

Use the CMSSERV command to start Enhanced Connectivity Facilities communications between your VM/SP host system and your work station (personal computer).

Format

CMSSERV	[(options... [])]
	<u>options:</u> [CUT DFT]

Options

CUT

specifies that CMSSERV is to be run in Control Unit Terminal mode.

DFT

specifies that CMSSERV is to be run in Distributed Function Terminal mode. The DFT option is the default. You can change the default mode using the DEFAULTS command.

Usage Notes

1. The CUT and DFT options cause the matching Communications Manager to be executed. If the called Communications Manager does not match the terminal mode, it displays an error message and prompts you to press the PF3 key to exit CMSSERV.
2. If the connection between your work station and the host virtual machine becomes disconnected while CMSSERV is running, then when you reconnect to your virtual machine, you may encounter unpredictable results. To recover the use of your virtual machine, you may need to re-ipl CMS. This can be accomplished by pressing the PA1 key, and typing IPL CMS, and pressing ENTER.
3. When you enter CMSSERV, the CMSSERV panel is displayed. If you enter CMSSERV and have not previously started your work station communications program, you'll be prompted to do so. From the CMSSERV panel, you can:
 - Swap (jump) to your work station (IBM Personal Computer) environment to use the services of Enhanced Connectivity Facilities.

You can also:

- If you are operating in DFT mode, enter any command on the command line that you can enter from CMS. There is no command line on the CUT mode panel; you cannot enter commands from the CUT mode panel.
- Press PF3 to end the communications program.

For more information about the services of Enhanced Connectivity Facilities, see *Introduction to IBM System/370 to IBM Personal Computer Enhanced*

Connectivity Facilities, GC23-0957 or VM/SP Programmer's Guide to the Server-Requester Programming Interface for VM/SP, SC24-5291.

4. If you want to issue CMSSERV from an exec, you should precede it with the EXEC command; that is, specify

```
exec cmsserv
```

5. If you do not have CUT Mode ECF installed on your work station and you attempt to start CMSSERV CUT mode from your host screen, "MORE..." or "HOLDING..." appears at the bottom of your screen. When you press the CLEAR key, this message appears:

```
DMSCUT1126S This terminal does not support CUT mode ECF.
           Press PF3 to end CMSSERV.
```

This means you have attempted to start CMSSERV, but your work station does not have CUT Mode ECF installed. CMSSERV is not active, but you must press PF3 to return.

If this happens:

- Verify that you have ECF CUT mode support installed on your work station and that it is correctly installed.
- If you have DFT Mode ECF installed on your work station, change the default CMSSERV option to DFT by issuing:

```
defaults set cmsserv dft
```

 Then try starting CMSSERV again.
- If your work station does not have any IBM ECF program installed and you want to use IBM ECF, obtain the correct work station ECF program and install it. See the *Introduction to IBM System/370 to IBM Personal Computer Connectivity Facilities, GC23-0957.*

Messages

DMSDFT639E	Error in <i>routine</i> routine; return code was <i>nnnn</i>
DMSDFT716E	SRPI subcommand environment was not found.
DMSDFT718E	Unable to link to work station.
DMSDFT719E	Work station communications not active.
DMSDFT720E	No longer linked to work station; error code was <i>nnn</i>
DMSDFT812E	Input was ignored.

Return Codes

0	Normal termination.
38	CMSSERV was called to start communications, but CMSSERV communications were already started.
40	General internal CMS error (for example, DMSDFT not a nucleus extension or error trying to NUCXLOAD DMSDFT).
55	Enhanced Connectivity Facilities communications error.
100	Console usage error.
104	Storage error.
nnnn	Propagated return code from routine (accompanies message DMSDFT639E).

COMPARE

Use the COMPARE command to compare two CMS files on a record-for-record basis and to display dissimilar records at the terminal.

Format

COMpare	<i>fileid1</i> <i>fileid2</i> [(option... [])]
	Option: [COL <i>mmm</i> [<i>nnn</i>] COL <i>mmm-<u>nnn</u></i>]

Operands

fileid1 fileid2

are the file identifiers of the files to be compared. An equal sign may be coded for one or more of the file identifiers for *fileid2* in any combination except '= = ='. All three file identifiers (file name, file type, file mode) must be specified for each file ID. An equal sign (=) coded in *fileid2* implies that the file identifier in that position is identical to the corresponding file identifier in *fileid1*.

If the COL option is not specified, the entire records are compared, the first column through the last character of each record (LRECL).

Option

COL *mmm* [*nnn*]

COL *mmm-nnn*

defines specific columns to be compared. The comparison begins at column *mmm* of each record. The comparison proceeds up to and including column *nnn*. If column *nnn* is not specified, the default ending column is the logical record length (LRECL). Only the first eight characters are examined for each number, so *mmm* and *nnn* must each be eight characters or less.

If you use "*mmm-nnn*," do not put blanks between the numbers and the hyphen (-). Only the first eight characters are examined, so the phrase "*mmm-nnn*" must be eight characters or less.

Usage Notes

1. To find out whether two files are identical, enter both file identifications, as follows:

```
compare test1 assemble a test1 assemble b
```

or

```
compare test1 assemble a = = b
```

Any records that do not match are displayed at the terminal.

2. To stop the display of dissimilar records, use the CMS Immediate command HT.
3. If a file does not exist on a specified disk or directory, the read-only extensions are also searched. The complete file IDs of the files being compared are displayed in message DMSCMP179I.

COMPARE

Responses

DMSCMP179I Comparing *fn ft fm* with *fn ft fm*

This message identifies the files being compared. If the files are the same (in the columns indicated), this message is followed by the CMS ready message. If any records do not match, the records are displayed. When all dissimilar records have been displayed the message DMSCMP209W is issued.

Messages and Return Codes

DMSCMP003E Invalid option: *option* [RC = 24]
DMSCMP005E No COLUMN specified [RC = 24]
DMSCMP009E Column *col* exceeds record length [RC = 24]
DMSCMP010E Premature EOF on file *fn ft [fm]* [RC = 40]
DMSCMP011E Conflicting file formats [RC = 32]
DMSCMP019E Identical fileids [RC = 24]
DMSCMP029E Invalid parameter *parameter* in the option *option* field [RC = 24]
DMSCMP054E Incomplete fileid specified [RC = 24]
DMSCMP062E Invalid * in fileid [RC = 20]
DMSCMP104S Error *nn* reading file *fn ft fm* from disk or directory [RC = 100]
DMSCMP109S Virtual storage capacity exceeded [RC = 104]
DMSCMP209W Files do not compare [RC = 4]
DMSCMP211E Column fields out of sequence [RC = 24]
DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC = 31]
DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

CONVERT COMMANDS

Use the CONVERT COMMANDS command to convert a CMS file containing Definition Language for Command Syntax (DLCS) statements into an internal form for the parsing facility.

Format

CONVERT COMMANDS	$[fn \ [ft \ \underline{\text{DLCS}} \ [fm \ * \ - \]]] \ [(options... \ (\) \)]$ $\underline{\text{Options:}} \ [\ \underline{\text{SYSTEM}} \ \underline{\text{USER}} \ \underline{\text{ALL}} \] \ [\ \underline{\text{CHECK}} \ \underline{\text{OUTMODE}} \ [fm \ * \ - \] \]$ $[\ \underline{\text{STACK}} \ \underline{\text{FIFO}} \ \underline{\text{LIFO}} \]$
------------------	---

Operands

fn

is the file name of the file to be converted. When editing a file, omit the file name (and file type and file mode) to convert the active file.

ft

is the file type of the file to be converted. If you do not specify *ft*, the default of DLCS is assumed.

fm

is the file mode of the file to be converted. The default for *fm* is an asterisk (*), which means the first file in the search order that satisfies the file name and file type qualifications is converted.

Options

SYSTEM

specifies that only valid system functions and subsets are in the file. This is the default.

USER

specifies that valid user functions and/or system functions and their subsets are in the file. You must load user functions as a nucleus extension for CONVERT COMMANDS to process them correctly.

ALL

specifies that the file contains user functions or subset values for user functions. They are not checked for validity.

CHECK

checks only the contents of the DLCS file for validity and does not produce an output text deck.

CONVERT COMMANDS

OUTmode

specifies the file mode to which the converted output files are written. The *fm* may be any read/write disk or directory that you have accessed. If you omit this parameter or specify an asterisk (*), the default is the first read/write disk or directory in the disk search order. OUTMODE * is the default if CHECK or OUTMODE *fm* is not specified.

STACK [FIFO]

STACK LIFO

causes the file IDs of the output files to be placed in the program stack. The stacked line has the format:

fn1 ft1 fm1 fn2 ft2 fm2

The *fn1 ft1 fm1* is the file ID of the file to be converted (DLCS file). The *fn2 ft2 fm2* is the file ID of the output file containing translations.

The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO

(first-in first-out) is the default option for STACK. FIFO causes the file IDs of the output files to be placed in the program stack. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

(last-in first-out) causes the file IDs of the output files to be placed in the program stack. This option is equivalent to STACK LIFO.

CONVERT COMMANDS always renames the text decks that the parser has created for you and gives them a new file type of TEXT. The file name of the output parser and synonym object files is determined by the contents of the DLCS statement in the source DLCS file. The output file name will be in one of two forms depending upon whether the SYSTEM or USER option is specified on the DLCS statement in the file to be converted.

1. System Parser and Synonym Files
 - xxxSPAcc TEXT
 - xxxSSYcc TEXT
2. User Parser and Synonym Files
 - xxxUPAcc TEXT
 - xxxUSYcc TEXT

where:

xxx

is the three-character application ID.

S or U

indicates whether the output file is a SYSTEM file or USER file.

PA

indicates that the file contains command syntax information.

SY

indicates that the file contains command name translations and synonyms.

cc

is the one or two-character country code for the national language you are using.

Note: Four utility files, COMMANDS CMSUT1, COMMANDS CMSUT2, COMMANDS ASSEMBLE, and COMMANDS TEXT are temporarily created to hold the converted data before it is placed in the output files.

Usage Notes

1. For detailed usage information, refer to the *VM/SP Application Development Guide for CMS* publication.
2. When editing a file with DLCS statements, you can issue CONVERT COMMANDS for that file from the XEDIT command line. If an error is detected, the line containing the error becomes the current line of the file.
3. System function subsets are always verified and cause an error when incorrect.
4. If both OUTmode and CHECK are specified, the one specified last is recognized.
5. If STACK, LIFO, or FIFO is specified with the CHECK option, a blank line is stacked.
6. Specifying LIFO or FIFO overrides STACK.
7. If conflicting options are specified, the last one specified is used.
8. If you want to issue CONVERT COMMANDS from an EXEC program, you should precede it with the EXEC command; that is, specify

```
exec convert commands
```

Example

Suppose you have a CMS file called TEST1 DLCS which uses AMENG (American English), and the file has the following statements:

```
:DLCS DMS USER AMENG ;;
:CMD MMYCMD1 MYCMD1 ;;
:SYN MY1 3 ;;
:OPR FCN(FN) ;;
:OPR FCN(FT) ;;
:OPT KWL(<DISK 4> <PRINT 5>) ;;
:OPT KWL(<NUMRECS 3>) FCN(PINTEGER) ;;
:CMD YYOURCMD YOURCMD YOURCMD 4 ;;
:OPR FCN(STRING) ;;
:OPT KWL(<TYPE 4>) ;;
```

To convert your file into a format that can be used by SET LANGUAGE and the parsing facility, enter:

```
convert commands test1 dlcs (system
```

The output files are DMSUPA TEXT and DMSUSY TEXT.

Suppose you have a second CMS file called TEST2 DLCS which uses UCENG (upper case English) instead of AMENG, and the file has the following statements:

CONVERT COMMANDS

```
:DLCS DMS USER UCENG ;;
:CMD MMYCMD1 MYCMD1 ;;
:SYN MY1 3 ;;
:OPR FCN(FN) ;;
:OPR FCN(FT) ;;
:OPT KWL(<DISK 4> <PRINT 5>) ;;
:OPT KWL(<NUMRECS 3>) FCN(PINTEGER) ;;
:CMD YYOURCMD YOURCMD YOURCMD 4 ;;
:OPR FCN(STRING) ;;
:OPT KWL(<TYPE 4>) ;;
```

To convert your file into a format that can be used by SET LANGUAGE and the parsing facility, enter:

```
convert commands test2 dlcs (system
```

The output files are DMSUPAB TEXT and DMSUSYB TEXT.

Messages and Return Codes

DMSCON006E	No read/write filemode accessed [RC = 36]
DMSPCON639E	Error in <i>routine</i> routine; return code was <i>retcode</i> [RC = 256]
DMSCON950I	Conversion of <i>fn ft fm</i> [from XEDIT] complete
DMSCON950I	No errors found in <i>fn ft fm</i> [from XEDIT]
DMSPCA947E	Line <i>line</i> : <i>syntax error</i> [RC = 8]
DMSPCC946E	XEDIT is not active. Specify a file name. [RC = 40]
DMSPCR002E	File <i>fn ft fm</i> not found [RC = 28]
DMSPCR069E	Filemode <i>fm</i> not accessed [RC = 36]
DMSPCR948E	Line <i>line</i> : <i>syntax error</i> [RC = 8]
DMSPCR949E	Line <i>line</i> : <i>syntax error</i> [RC = 8]
DMSPCT396E	Maximum number of command table entries exceeded [RC = 32]
DMSPCT639E	Error in <i>routine</i> routine; return code was <i>retcode</i> [RC = 256]
DMSPCW396E	Maximum number of command table entries exceeded [RC = 32]
DMSPCW639E	Error in <i>routine</i> routine; return code was <i>retcode</i> [RC = 256]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813
Errors in copying a file	70

CONWAIT

Use the CONWAIT command to cause a program to wait until all pending terminal I/O is complete.

Format

CONWAIT	
---------	--

Usage Note

The CONWAIT command synchronizes input and output to the terminal; it ensures that the output console stack is cleared before the program continues execution. Also, you can ensure that a read or write operation is finished before you modify an I/O buffer.

Options

Type

displays, at the terminal, the names of the files being copied.

NOType

suppresses the display of the names of the files being copied.

NEWDate

uses the current date as the creation date of the new file(s).

OLDDate

uses the date on the (first) input file as the creation date of the new file(s).

NEWFile

checks that files with the same file ID as the output file do not already exist. If one or more output files do exist, an error message is displayed and the COPYFILE command terminates. This option is the default so that existing files are not inadvertently destroyed.

REPlace

causes the output file to replace an existing file with the same file identifier. REPLACE is the default option when only one file ID is entered or when the output file ID is specified as “= = =.”

PRompt

displays the messages that request specification or translation lists.

NOPRompt

suppresses the display of prompting messages for specification and translation lists.

Copy Extent Options

FFrom *recno*

is the starting record number for each input file in the copy operation.

FRLabel *xxxxxxxx*

xxxxxxxx is a character string that appears at the beginning of the first record to be copied from each input file. Up to eight non-blank characters may be specified.

FOR *numrec*

is the number of records to be copied from each input file.

TOLabel *xxxxxxxx*

xxxxxxxx is a character string which, if at the beginning of a record, stops the copy operation for that input file. The record containing the given character string is not copied. Up to eight non-blank characters may be specified.

SPecs

indicates that you are going to enter a specification list to define how records should be copied. See “Entering a COPYFILE Specification List” on page 68 for information on how you can define output records in a specification list.

NOSPecs

indicates that no specification list is to be entered.

OVly

overlays the data in an existing output file with data from the input file. You can use OVLY with the SPECS option to overlay data in particular columns.

APpend

places information at the end of the output file.

Data Modification Options

The following options can be used to change the record format of a file. See "Modifying Record Formats" on page 67 for more details.

RECFm F**RECFm V**

is the record format of the output files. If not specified, the output record format is the same as that of the input file.

LRecl *nnnn*

is the logical record length of the output file(s) if it is to be different from that of the input file(s). The maximum value of *nnnn* is 65535.

TRUnc

removes trailing blanks (or fill characters) when converting fixed-length files to variable-length format.

NOTRunc

suppresses the removal of trailing blanks (or fill characters) when converting fixed-length files to variable-length format.

PAck

compresses records in a file so that they can be stored in packed format.

Note: A file in packed format should not be modified in any way. If such a file is modified, the UNPACK routines are unable to reconstruct the original file.

UNPack

reverses the PACK operation. If a file is inadvertently packed twice, you can restore the file to its original unpacked form by issuing the COPYFILE command twice.

Fll *c***Fll *hh*****Fll 40**

is the padding and truncation character for the TRUNC option or the principal packing character for the PACK option. The fill character may be specified as a single character, *c*, or by entering a two-digit hexadecimal representation of a character. The default is 40 (the hexadecimal representation for a blank in EBCDIC).

Character Translation Options**EBcdic**

converts a file that was created with 026 keypunch characters (BCD), to 029 keypunch characters (EBCDIC). The following conversions are made:

```

< to )
& to +
% to (
# to =
@ to '
' to :

```

UPcase

converts all lowercase characters in each record to uppercase before writing the record to the output file.

LOWcase

converts all uppercase characters in each record to lowercase before writing the record to the output file.

TRAns

indicates that you are going to enter a list of character translations to be made as the file is copied. See "Entering Translation Specifications" on page 69 for details on entering a list of characters to be translated.

Single

suppresses multiple output mode regardless of how the file identifiers are specified.

Incompatible Options

Table 6 below shows combinations of options that should *not* be specified together in the same COPYFILE command. If the option in the first column is specified, do not code any of the options in the second column.

Options	Incompatible Options
APPEND	LRECL, NEWDATE, NEWFILE, OLDDATE, OVLY, PACK, RECFM, REPLACE, UNPACK
EBCDIC	PACK, UNPACK
FOR	PACK, TOLABEL, UNPACK
FRLABEL	FROM, PACK, UNPACK
FROM	FRLABEL, PACK, UNPACK
LOWCASE	PACK, UNPACK
LRECL	APPEND, PACK, UNPACK
NEWDATE	APPEND, OLDDATE
NEWFILE	APPEND, OVLY, REPLACE
NOPROMPT	PROMPT
NOSPECS	SPECS
NOTRUNC	TRUNC
NOTYPE	TYPE
OLDDATE	APPEND, NEWDATE
OVLY	APPEND, NEWFILE, PACK, REPLACE, UNPACK
PACK	APPEND, EBCDIC, FOR, FRLABEL, FROM, LOWCASE, LRECL, OVLY, RECFM, SPECS, TOLABEL, TRANS, TRUNC, UNPACK, UPCASE
PROMPT	NOPROMPT
RECFM	APPEND, PACK, UNPACK
REPLACE	APPEND, NEWFILE, OVLY
SPECS	NOSPECS, PACK, UNPACK
TOLABEL	FOR, PACK, UNPACK
TRANS	PACK, UNPACK
TRUNC	NOTRUNC, PACK, UNPACK
TYPE	NOTYPE
UNPACK	APPEND, EBCDIC, FOR, FRLABEL, FROM, LOWCASE, LRECL, OVLY, PACK, RECFM, SPECS, TOLABEL, TRANS, TRUNC, UPCASE
UPCASE	PACK, UNPACK

COPYFILE

Using the COPYFILE Command

Two simple uses of the COPYFILE command are:

1. to copy a single CMS file from one minidisk to another, from one SFS directory to another, or between minidisks and directories,
2. to make a duplicate copy of the file on the same minidisk or directory.

For example:

```
copyfile test1 assemble a test2 assemble a
```

makes a copy of the file TEST1 ASSEMBLE A and names it TEST2 ASSEMBLE A.

For those portions of the file identifier that you want to stay the same, you may code an equal sign in the output file ID. Thus, the command line above can be entered:

```
copyfile test1 assemble a test2 = =
```

The equal sign may be used as a prefix or suffix of a file identifier. For example, the command:

```
copyfile a b c file= type= =
```

creates an output file called FILEA TYPEB C.

When you copy a file from one disk or directory to another, you specify the old and new file modes, and any file name or file type change you want to make; for example:

```
copyfile test3 assemble c good = a
```

This command makes a copy of the file TEST3 ASSEMBLE C, and names it GOOD ASSEMBLE A.

You can change the file mode number of a CMS file by using the COPYFILE command with the REPLACE option. For example,

```
copyfile test assemble a1 = = a4 (replace
```

Note: If you change the file mode number of a base file, the file mode number of all the aliases associated with the base file are changed also. The same applies if you change the file mode number of an alias - the file mode for the base file gets changed as well as all aliases.

If you want to copy only particular records in a file, you can use the FROM/FOR FRLABEL/TOLABEL options. For example:

```
copyfile old test a new test a (frlabel start for 41
```

copies 41 records from the file OLD TEST A1, beginning with the record starting with the character string START into the file NEW TEST A1. Since the user's command line, as passed to COPYFILE in the PLIST, has been translated into uppercase letters, any FRLABEL or TOLABEL character string consisting of either all lowercase or mixed case letters is not found in the input file. Error message DMSCPY157E is issued if the FRLABEL character string is not found. If the TOLABEL character string is not found, the copy operation continues as if TOLABEL was not specified.

Note: If the input file mode is an '*', then you should specify an explicit file mode, not an '=', for the output file mode. If you do not specify an explicit output file

mode, it is possible to create an output file that would be recognized as an input file which generates the error message DMSCPY024E stating that the file already exists. For example, if you have a file named 'C B A' and you issue the command 'COPY C * * = D =', COPY will first create an output file named 'C D A'. This file will then match the input file ID of the file 'C * *' and copy will attempt to write an output file with the name 'C D A', which already exists.

Copying Files in SFS Directories

To copy files in SFS directories, the directories must first be accessed (see the ACCESS command). New files created by using the COPYFILE command have none of the original file's authorities or aliases.

You can use the COPYFILE command with the REPLACE option to replace any file for which you have write authority, even if the file is in another user's directory that you have accessed as read only. The authorities and aliases that existed for the file prior to replacing it remain in effect. For example, you want to replace a file, PRINT ASSEMBLE, in Tom's directory with your file named TEST ASSEMBLE. You have write authority to PRINT ASSEMBLE and you have accessed Tom's directory as B. Issuing,

```
copyfile test assemble a print = b (replace
```

replaces PRINT ASSEMBLE with the contents of TEST ASSEMBLE. All of the authorities and aliases on the original PRINT ASSEMBLE remain in effect after it has been replaced. In the above example, if PRINT ASSEMBLE is an alias for a base file named PD77MSHC ASSEMBLE, the contents of PD77MSHC ASSEMBLE are replaced.

If you have write authority for another user's directory, you can use COPYFILE to create a new file in that user's directory even though the directory is accessed as read only. The owner of the directory becomes the owner of the file. You, as the creator of the file, automatically have write authority to it.

You cannot copy a file to a file or directory (*fileido*) that you have locked SHARE; or to a file or directory that another user has locked SHARE, UPDATE, or EXCLUSIVE. You cannot copy a file (*fileid1*) that is locked EXCLUSIVE by another user.

Multiple Input and Output Files

You can combine two or more files into a single file with the COPYFILE command. For example:

```
copyfile test data1 a test data2 = test data3 b
```

copies the files TEST DATA1 and TEST DATA2 from your disk or directory accessed as A and combines them into a file, TEST DATA3, on your disk or directory accessed as B.

Note that if any input file has a file mode number of 3, it is possible that the file will be copied in a sequence different from its order on the disk or directory.

If you want to combine two more files without creating a new file; use the APPEND option. For example:

```
copyfile new list a old list a (append
```

appends the file NEW LIST A to the bottom of the existing file labeled OLD LIST A.

COPYFILE

Note: If the file NEW LIST A has a different LRECL from the file OLD LIST A, the appended data is padded, or truncated, to the LRECL of the file OLD LIST A.

Whenever you code an asterisk (*) in an input file ID, you may cause one or more files to be copied, depending upon the number of files that satisfy the remaining conditions. For example:

```
copyfile * test a combined test a
```

copies all files with a file type of TEST into a single file named COMBINED TEST. If only one file with a file type of TEST exists, only that file is copied.

If you want to copy all the files on a particular disk or directory to another disk or directory, you could enter:

```
copyfile * * b = = a
```

All the files on the disk or directory accessed as B are copied to the one accessed as A. The file names and file types remain unchanged.

You can also copy a group of files and change all the file names or all the file types. For example:

```
copyfile * assemble b = test a
```

copies all ASSEMBLE files on the disk or directory accessed as B into files with a file type of TEST on the disk or directory accessed as A. The file names are not changed.

You can use the SINGLE option to override multiple output mode. For example:

```
copyfile * test a = = B (single
```

copies all files on the disk or directory accessed as A with a file type of TEST to the disk or directory accessed as B as one combined file, with the file name and file type equal to the first input file found.

Whenever an asterisk appears, it indicates that all files are to be copied; whenever an equal sign (=) appears, it indicates that the same files are to be copied. For example:

```
copyfile x * a1 = file =
```

combines all files with a file name of X on the disk or directory accessed as A into a single file named X FILE A1.

Whenever an equal sign appears in the output file ID in a position corresponding to an asterisk in an input file ID, multiple input files produce multiple output files. When you perform copy operations of this nature you might wish to use the TYPE option, which displays the names of files being copied. For example:

```
copyfile * test a = output a = summary = (type
```

might result in the display:

```
COPY 'ALPHA TEST A1' TO 'ALPHA SUMMARY A1' (NEW FILE)
COPY 'ALPHA OUTPUT A'
COPY 'BETA TEST A1' TO 'BETA SUMMARY A1' (NEW FILE)
COPY 'BETA OUTPUT A.'
```

which indicates that files ALPHA TEST A and ALPHA OUTPUT A were copied into a file named ALPHA SUMMARY A and that files BETA TEST A and BETA OUTPUT A were copied into a file named BETA SUMMARY A.

Modifying Record Formats

You can use the RECFM and LRECL options to change the record format of a file as you copy it. For example:

```
copyfile data file a (recfm f lrecl 130
```

converts the file DATA FILE A1 to fixed-length 130-character records.

If you specify an output file ID, for example:

```
copyfile data file a fixdata file a (recfm f lrecl 130
```

the original file remains unchanged. The file FIXDATA FILE A contains the converted records.

If the records in a file being copied are variable-length, each output record is padded with blanks to the specified record length. If any records are longer than the record length, they are truncated.

When you convert files from fixed-length records to variable-length records, you can specify the TRUNC option to ensure that all trailing blanks are truncated:

```
copyfile data file a (recfm v trunc
```

If you specify the LRECL option and RECFM V, the LRECL option is ignored and the output record length is taken from the longest record in the input file.

When you convert a file from variable-length to fixed-length records, you may also specify a fill character to be used for padding instead of a blank. If you specify:

```
copyfile short recs a (recfm f fill *
```

then each record in the file SHORT RECS is padded with asterisks to the record length. Assuming that SHORT RECS was originally a variable-length file, the record length is taken from the longest existing record. Note that if SHORT RECS is already fixed-length, it is not altered.

Similarly, when you are converting back to variable-length a file that was padded with a character other than a blank, you must specify the FILL option to indicate the pad character, so that character is truncated.

The FILL option can also be used to specify the packing character used with the PACK option. When you use the PACK option, a file is compressed as follows: all occurrences of two or more blanks are encoded as one character, and four or more occurrences of any other character are written as three characters. If you use the FILL option to specify a fill character, then that character is treated as a blank when records are compressed. You must, of course, specify the FILL option to unpack any files packed in this way. Since most fixed-length files are blank-padded to the record length, you do not need to specify the FILL option unless you know that some other character appears more frequently.

A file which is packed on an 800 byte blocksize disk will be fixed format with a logical record length of 800. On a 512, 1K, 2K, or 4K blocksize disk, the file will be fixed format with a logical record length of 1024. A packed file of either logical record length can be unpacked back to its original specifications regardless of the disk blocksize it resides on. A packed file with logical record length 800 on a disk with blocksize 512, 1K, 2K, or 4K, and packed files with logical record length 1024 on 800 byte disks should be unpacked and re-packed if minimal disk block usage is needed.

COPYFILE

When you convert record formats on packed files with the COPYFILE command you can specify single or multiple output files, as outlined in the previous section, "Modifying Record Formats" on page 67. For example:

```
copyfile * assemble a (pack
```

compresses all ASSEMBLE files in the disk or directory accessed as A without changing any file identifiers. The command:

```
copyfile * assemble a = script = (recfm v trunc
```

creates copies of all ASSEMBLE files residing on your disk or directory accessed as A. The copies will have variable-length record formats and file types of SCRIPT.

Entering a COPYFILE Specification List

When you use the COPYFILE command, you can specify particular columns of data to be manipulated or particular characters to be translated. Again, how you specify the file identifier determines how many files are copied or modified.

When you use the SPECS option on the COPYFILE command, you receive the message:

```
DMSCPY601R      Enter specification list:
```

The system waits for you to enter a specification list. If you do not wish to receive this message, use the NOPROMPT option. The specification list you enter may consist of one or more pairs of operands in the following format:

```
nnn-mmm  
/string/   col  
hxx...
```

where:

nnn-mmm

specifies the start and end columns of the input file that are to be copied to the output file. If mmm exceeds the length of the input record, the end of the record is the assumed ending position.

string

is any string of uppercase and lowercase characters or numbers delimited by any non-alphanumeric character.

hxx...

is an even number of hexadecimal digits prefixed with an h.

col

is the column in the output file at which the copy operation is to begin.

You can enter as many as 20 pairs of specifications resulting in as many as 130 characters per line. If you want to enter more than one line of specifications, enter two plus signs (+ +) before column 130 at the end of one input line as continuation indicators.

A specification list may contain any combination of specification pairs; for example:

```
copyfile sorted list a (specs
```

```
DMSCPY601R    Enter specification list:
```

```
/|/ 1 1-8 3 /|/ 12 /***/ 14 ++
9-80 18
```

After this command is executed, each record in the file SORTED LIST will look like the following:

```
| 0000000 | *** 0000....
```

where the o's in columns 3 through 10 indicate information originally in columns 1 through 8; the o's following the asterisks indicate the remainder of each record, columns 9 through 80.

When you enter a specification list, you are actually constructing a file column by column. If you specify multiple input or output files, the same copy operation is performed for each record in each output file.

Those columns for which you do not specify any data are filled with blanks or, if you use the FILL option, the fill character of your choice. For example:

```
copyfile sorted list a (specs noprompt 1recl 20 fill $
1-15 6
```

copies columns 1 through 15 beginning in column 6 and writes dollar signs(\$) in columns 1 through 5.

If you do want to modify data in particular columns of a file but want to leave all of the rest of each record unchanged, you can use the OVLY (overlay) option. For example, the sequence:

```
COPYFILE * bracket a (specs ovly noprompt
had 1 hbd 80
```

overlays the characters [(X'AD') and] (X'BD') in columns 1 and 80 of all the files with a file type of BRACKET on your disk or directory accessed as A.

When you copy fixed-length files, records are padded or truncated to the record length; variable-length files are always written as specified.

Entering Translation Specifications

You can perform conversion on particular characters in CMS files or groups of files with the TRANS option of the COPYFILE command.

When you enter the TRANS option, you receive the message:

```
DMSCPY602R    Enter translation list:
```

and a read is presented to your virtual machine. You may enter the translation list. If you do not wish to receive this message, use the NOPROMPT option.

A translation list consists of one or more pairs of characters or hex digits, each pair representing the character you want to translate and the character you want to translate it to, respectively. For example:

```
copy test file a (trans
```


DMSCPY602R Enter translation list:

* - A f0 00 ff

specifies that all occurrences of the character * are to be translated to -, all character A's are to be translated to X'F0' and all X'00's are to be translated to X'FF's.

If any translation specifications you enter conflict with the LOWCASE, EBCDIC, or UPCASE options specified on the same command line, the translation list takes precedence. In the preceding example, if LOWCASE had also been specified, all A's would be translated to X'F0's, not to a's.

You can enter as many as 130 characters per line. You can enter translation pairs on more than one line if you enter two plus signs (++) before column 130 at the end of one input line as continuation indicators.

Responses

DMSCPY601R Enter specification list:

This message prompts you to enter a specification list when you use the SPECS option.

DMSCPY602R Enter translation list:

This message prompts you to enter a translation list when you use the TRANS option.

DMSCPY721I Copy *fn ft fm* {to|append|overlay} *fn ft fm* ({old|new} file)

This message appears for each file copied with the TYPE option. It indicates the names of the input file and output file. When you have multiple input files, the output file ID is displayed only once.

Messages and Return Codes

DMSCPY002E File(s) [*fn* [*ft* [*fm*]]] not found [RC = 28]
 DMSCPY002E Input file(s) [*fn* [*ft* [*fm*]]] not found [RC = 28]
 DMSCPY002E Overlay file(s) [*fn* [*ft* [*fm*]]] not found [RC = 28]
 DMSCPY003E Invalid option: *option* [RC = 24]
 DMSCPY024E File *fn ft fm* already exists; specify REPLACE option [RC = 28]
 DMSCPY029E Invalid parameter *parameter* in the option *option* field [RC = 24]
 DMSCPY030E File *fn ft fm* already active [RC = 28]
 DMSCPY037E Filemode *mode*(*vdev*) is accessed as read/only [RC = 36]
 DMSCPY042E No fileid(s) specified [RC = 24]
 DMSCPY048E Invalid mode *mode* [RC = 24]
 DMSCPY054E Incomplete fileid specified [RC = 24]
 DMSCPY062E Invalid character *char* in fileid *fn ft fm* [RC = 20]
 DMSCPY063E No [sort|translation|specification] list {entered|given} [RC = 40]
 DMSCPY064E Invalid [translate] specification at or near *list* [RC = 24]
 DMSCPY065E *option* option specified twice [RC = 24]
 DMSCPY066E *option1* and *option2* are conflicting options [RC = 24]
 DMSCPY067E Combined input files illegal with PACK or UNPACK options [RC = 24]
 DMSCPY068E Input file *fn ft fm* not in packed format [RC = 32]
 DMSCPY069E Filemode *mode* not accessed [RC = 36]
 DMSCPY101S SPECS temp string storage exhausted at *storarea* [RC = 88]
 DMSCPY102S Too many fileids [RC = 88]
 DMSCPY103S Number of SPECS exceeds maximum *nn* [RC = 88]
 DMSCPY104S Error *nn* reading file *fn ft fm* [RC = 100]

DMSCPY105S Error *nn* writing file *fn ft fm* on disk or directory [RC = 100]
 DMSCPY156E FROM *nnn* not found--the file *fn ft fm* has only *nnn* records
 [RC = 32]
 DMSCPY157E Label *label* not found in file *fn ft fm* [RC = 32]
 DMSCPY172E TOLABEL *label* {equals|is an initial substring of} FRLABEL
label [RC = 24]
 DMSCPY173E No records were copied to output file *fn ft fm* [RC = 40]
 DMSCPY901T Unexpected error at *vstor1*: *plist function fn ft fm* at *vstor2*, base
vstor3, rc *nn* [RC = 256]
 DMSCPY903T Impossible PHASE code *xx* [RC = 256]
 DMSCPY904T Unexpected UNPACK error at *vstor1*, base *vstor2* [RC = 256]
 DMSCPY1184E File *fn ft fm* not found or you are not authorized for it [RC = 28]
 DMSCPY1258E You are not authorized to write to file *fn ft fm* [RC = 76]
 DMSCPY1259E File pool *filepoolid* has run out of physical space in the storage
 group [RC = 40]
 DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC = 31]
 DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

CP

CP

Use the CP command to transmit commands to the VM/SP control program environment without leaving the CMS environment.

Format

CP	[<i>commandline</i>]
----	------------------------

Operands

commandline

is any CP command valid for your CP command privilege class. If this field is omitted, you are placed in the CP environment and may enter CP commands without preceding each command with CP. To return to CMS, issue the CP command BEGIN.

Usage Notes

1. You must use the CP command to invoke a CP command:
 - From within a CMS EXEC or an EXEC 2 EXEC.
 - If the implied CP (IMPCP) function is set to OFF for your virtual machine.
 - In a job that you send to the CMS batch facility.
2. To enter a CP command from the CMS environment without CMS processing the command line, use #CP.
3. When you enter an invalid CP command following the CP command, you receive a return code of -1. In an EXEC, this return code is +1.

Example

If the implied CP function is set to OFF for your virtual machine (QUERY IMPCP to find out setting), and you want to specify the CP SCREEN command, you can precede it by the CP command. For example:

```
cp screen inarea red
```

Responses

All responses are from the CP command that was issued; the CMS ready message follows the response.

CREATE ALIAS

Use the CREATE ALIAS command to create an additional name (*alias*) for a file in a Shared File System (SFS) directory. The alias may be placed in any directory for which you have write authority.

Once an alias has been created, the alias name becomes a pointer to the base file that contains the data. The alias does not contain data of its own. Use QUERY ALIAS to display alias information.

Format

CREate ALias	$\left\{ \begin{array}{c} fn1 \\ * \end{array} \right\} \left\{ \begin{array}{c} ft1 \\ * \end{array} \right\} \text{ dirid1 } \left\{ \begin{array}{c} fn2 \\ = \end{array} \right\} \left\{ \begin{array}{c} ft2 \\ = \end{array} \right\} \left\{ \begin{array}{c} dirid2 \\ = \end{array} \right\} [(\text{options...} [])]$ <p>Options:</p> $\left[\begin{array}{l} \text{TYPE} \\ \text{NOType} \\ \text{STACK} \left[\begin{array}{c} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{LIFO} \\ \text{FIFO} \end{array} \right]$
---------------------	---

Operands

fn1 ft1

identifies the file for which you are creating an alias. Special characters (* and %) can be used to designate a set of files. For information on using these special characters, refer to "Pattern Matching" on page 8. If you do use a special character, you must use an equal sign (=) for the corresponding *fn2* or *ft2*. For example:

```
create alias * assemble .proj1 = = .proj2
```

This would create aliases in your .PROJ2 directory for all the ASSEMBLE files in your .PROJ1 directory.

Only those files for which you have read or write authority will have aliases created.

dirid1

identifies the SFS directory that contains the file(s) on which you are creating the alias. For a more detailed description of *dirid*, refer to "Naming Shared File System (SFS) Directories" on page 4.

fn2 ft2

identifies the alias name you are creating. You may use equal signs (=) to specify that the *fn2* or *ft2* for the alias will be the same as the *fn1* or *ft1*.

dirid2

identifies the SFS directory in which you wish to place the alias. You have two choices for *dirid2*:

CREATE ALIAS

1. You can specify an equal sign (=) for *dirid2* which means the alias is placed in the same directory as *dirid1*. The alias (fn2 ft2) must be different than *fn1 ft1*. You must have write authority on the directory to do this.
2. You can specify a directory name for *dirid2* for which you have write authority.

If you specify *dirid2* as a file mode letter, do not use a file mode number. The file mode number of an alias is automatically assigned the same file mode number as the base file.

Options

TYPE

displays the alias name(s) at the terminal.

NOType

suppresses the display of the alias name(s) you create. NOType is the default.

STACK [FIFO]

STACK LIFO

places the output in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the output is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the output is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

Usage Notes

1. The new file is known as an alias while the original file is known as the base file. You can create an alias on an alias. The alias of an alias is really an alias of the base file.
2. Three conditions must be met to successfully create an alias:
 - You must have read or write authority to the base file.
 - You must have write authority to the target directory (*dirid2*).
 - The target directory owner must have read or write authority to the base file.
3. If USER A wants to share your file, you grant USER A read or write authority on the file. USER A can then issue CREATE ALIAS to that file in any owned directories or in directories on which USER A has write authority.

An alternate way to share the file is: USER A could grant you write authority on a particular directory. You could then create an alias for the file in USER A's directory.
4. If the owner of the base file revokes your authority to the file, any aliases you have to that base file will be changed to revoked aliases.
5. If the owner of the base file erases the base file, any aliases you have to that base file will be changed to erased aliases.
6. Aliases cannot be created for files located in another file pool or for files located on disks.
7. To create an alias on a file, the file cannot be open by another user.

8. If special characters (* or %) are used for *fn1* or *ft1*, the specified directory (*dirid1*) cannot be open.
9. If the base file is locked EXCLUSIVE, an alias can be created only by the lock holder. If the base file is locked SHARE or UPDATE, anyone with read or write authority to the file can create an alias for it.

If the target directory (*dirid2*) is locked EXCLUSIVE or UPDATE, only the lock holder can put an alias in it. If the target directory is locked SHARE, no one can put an alias in it.
10. To delete an alias, use the ERASE command or the DISCARD command from FILELIST.
11. You can use the RELOCATE command to move any aliases to another directory.
12. You may create as many aliases as you wish on the base file.
13. If special characters are used for *fn* or *ft*, aliases are not created in the specified directory for:
 - a. subdirectories
 - b. erased or revoked aliases
 - c. files that you are not authorized to either read or write.

Processing continues for the remaining file names that match the specified pattern.

14. You can issue the CREATE ALIAS command from the command line, from an exec, or as a function from a program. No error messages are issued if CREATE ALIAS is invoked:
 - As a function from a program
 - From a CMS exec file that has the &CONTROL NOMSG option in effect
 - From an EXEC2 exec where CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect

Examples

1. You may have a file in directory .PROJ1 which you want to reference through directory .PROJ2. To do this, issue CREATE ALIAS in directory .PROJ2 to the file in directory .PROJ1.

```
create alias cleanup exec .proj1 = = .proj2
```

This would create an alias in the .PROJ2 directory for the base file, CLEANUP EXEC.

2. You may also create an alias to a file in the same directory, thus creating a synonym for the file in the same directory.

```
create alias F67032 assemble .proj1 counter = =
```

This would create a synonym, COUNTER, in the same directory for the file, F67032.

CREATE ALIAS

Messages and Return Codes

DMSJAL002E File *fn ft fm|dirname* not found [RC=28]
DMSJAL1160E Directory *dirname* already open [RC=70]
DMSJAL1163E The CREATE ALIAS command failed for *fn ft fm|dirname*
[RC=nn]
DMSJAL1184E Directory *dirname* not found or you are not authorized for it
[RC=28]
DMSJAL1184E File *fn ft* or directory *dirname* not found [RC=28]
DMSJAL1184E File or directory not found or authorization requirements not met
[RC=28]
DMSJAL1210E Directory *dirname* not found [RC=28]
DMSJAL1241E Directories specified are in different file pools [RC=88]
DMSJAL1312E A filemode number may not be specified with the filemode of an
alias [RC=24]
DMSJED069E Filemode *mode* not accessed [RC=36]
DMSJED109S Virtual storage capacity exceeded [RC=104]
DMSJED1187E Too many subdirectory levels in *dirid* [RC=24]
DMSJED1188E Filemode *mode* is not associated with a directory [RC=74]
DMSJED1189E Filemode *mode* is associated with a top directory [RC=24]
DMSJED1223E There is no default file pool currently defined [RC=40]
DMSOUT109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

CREATE DIRECTORY

Use the CREATE DIRECTORY command to create a Shared File System (SFS) directory once you have been enrolled as a user in the Shared File System. To see a listing of directories, use DIRLIST or LISTDIR.

Format

CREate DIRectory	<i>dirid</i>
-------------------------	--------------

Operands

dirid

specifies the name of a new directory which you are creating. For this command, you cannot use *fm* alone as the *dirid* since this would not name a new directory. For a more detailed description of *dirid*, refer to "Naming Shared File System (SFS) Directories" on page 4.

Usage Notes

1. You cannot create a top directory using CREATE DIRECTORY. A top directory is automatically created when you are enrolled. Your user ID followed by a period is the name of the top directory.
2. You can create a maximum of nine levels of directories, including your top directory.
3. You can only create a directory structure one level at a time.
4. You cannot create a directory in another user's directory structure.
5. If a directory is locked, you cannot create a subdirectory within that directory. An exception is if you hold an UPDATE or EXCLUSIVE lock on the directory, then you can create a subdirectory within the directory.
6. You can invoke the CREATE DIRECTORY command from the command line, from an exec, or as a function from a program. No error messages are issued if CREATE DIRECTORY is invoked:
 - As a function from a program
 - From a CMS exec file that has the &CONTROL NOMSG option in effect
 - From an EXEC2 exec where CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect

Example

To create a subdirectory called COMPSCI under your top directory, enter:
create directory .compsci

To create a subdirectory named DATASTRUCT under COMPSCI:
create directory .compsci.datastruct

CREATE DIRECTORY

Messages and Return Codes

- DMSJCD1184E Directory *dirname* not found or you are not authorized to use CREATE DIRECTORY on this directory. [RC = 28]
- DMSJCD1218E You cannot create top directories using the CREATE DIRECTORY command [RC = 88]
- DMSJED109S Virtual storage capacity exceeded [RC = 104]
- DMSJED1187E Too many subdirectory levels in *dirid* [RC = 24]
- DMSJED1188E Filemode *mode* is not associated with a directory [RC = 74]
- DMSJED1189E Filemode *mode* is associated with a top directory [RC = 24]
- DMSJED1223E There is no default file pool currently defined [RC = 40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

CREATE LOCK

Use the CREATE LOCK command to create an explicit lock on a file or a directory. An explicit lock limits or prevents access to a directory or file. Use QUERY LOCK to display lock information.

Format

CREate LOCK	$\left[\begin{array}{cc} fn & ft \\ * & * \end{array} \right] \text{ dirid } \left\{ \begin{array}{l} \text{SHAre} \\ \text{EXClusive} \\ \text{UPDate} \end{array} \right\} \left\{ \begin{array}{l} \text{SESSion} \\ \text{LASTing} \end{array} \right\} [(\text{options...} [])]$ <p><u>Options:</u></p> $\left[\begin{array}{l} \text{TYPe} \\ \text{NOType} \\ \text{STACK } \left[\begin{array}{l} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{LIFO} \\ \text{FIFO} \end{array} \right]$
--------------------	--

Operands

fn ft

specifies the name of the file to be locked. You must have either read or write authority to lock a file depending on the type of lock you wish to create. Special characters (* and %) can be used to designate a set of files. See "Pattern Matching" on page 8 for more information on using these characters.

dirid

identifies the directory. If *fn* and *ft* are specified, this is the directory that contains the file to be locked. If *fn* and *ft* are not specified, this is the name of the directory to be locked. For a more detailed description of *dirid*, refer to "Naming Shared File System (SFS) Directories" on page 4.

SHAre

specifies that others may read while you read it. You must have read authority to use the SHARE operand.

EXClusive

prevents other users from modifying or reading regardless of the authority that may have been granted. You must have write authority to the base file or directory to use this operand.

UPDate

specifies that others may read while you read or modify. You must have write authority to use this operand.

SESSion

specifies that the lock is automatically removed when your CMS session with the file pool ends. A CMS session ends when any of the following occur:

- system reset

CREATE LOCK

- re-IPL
- file system server abend
- network or APPC/VM failure on the last communication link with a file pool
- log off

A DELETE LOCK command will remove the lock during the CMS session.

LASTing

specifies that the lock stays in effect until a DELETE LOCK command is issued. The lock remains across CMS sessions.

Options

TYPE

displays the names of the files for which you have created locks.

NOType

specifies that file information is not displayed at the terminal. NOTYPE is the default.

STACK [FIFO]

STACK LIFO

places the information in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

Usage Notes

1. If special characters are used for *fn* or *ft*, locks are not created in the directory for:
 - a. subdirectories
 - b. erased or revoked aliases
 - c. files that you are not authorized to read or write.

Processing continues for the remaining file names that match the specified pattern.

2. If another user has an UPDATE or EXCLUSIVE lock on a file or directory, you cannot create a lock on the same file or directory. Multiple SHARE locks are allowed. If you have a lock on a file or directory, you must delete the lock before creating another lock on it. See the DELETE LOCK command.

Existing locks can affect whether or not you can create a lock on a file or directory.

- You can always lock a directory that contains locked files if you hold the locks, unless you are trying to lock the directory SHARE and you have one or more files locked UPDATE or EXCLUSIVE. If another user has files locked, you cannot lock the directory, unless the locks are all SHARE and you are trying to lock the directory SHARE or UPDATE.

- You can always lock files in a locked directory if you hold the lock, unless you are trying to lock the files UPDATE or EXCLUSIVE and you have the directory locked SHARE. If another user has the directory locked, you cannot lock files in that directory, unless the directory is locked SHARE or UPDATE and you are trying to lock the files SHARE.
3. If special characters (* and %) are used for *fn* or *ft*, the specified directory cannot be open.
 4. If you create an EXCLUSIVE lock on a directory and another user already has the directory accessed, their access to the directory remains in effect. Even if the user releases the directory, the user will be able to reaccess the directory until the user re-IPLs or their connection to the file pool is broken.
 5. When using the batch machine, it runs the job under your user ID, so
 - Even if you have an EXCLUSIVE lock on a file, your job running on the batch machine can use the file.
 - If your batch machine has a file open to update it, then any job running from your own virtual machine cannot update the file.
 6. See the SET FILEWAIT command for information on how the FILEWAIT setting affects the CREATE LOCK command.
 7. If the CREATE LOCK command is issued from an exec on a work unit that has not been committed, the command will fail.
 8. You can invoke the CREATE LOCK command from the command line, from an exec, or as a function from a program. No error messages are issued if CREATE LOCK is invoked:
 - As a function from a program
 - From a CMS exec file that has the &CONTROL NOMSG option in effect
 - From an EXEC2 exec where CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect

Messages and Return Codes

DMSJCR1132E	Invalid number of operands [RC = 24]
DMSJED069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSJED109S	Virtual storage capacity exceeded [RC = 104]
DMSJED1187E	Too many subdirectory levels in <i>dirname</i> [RC = 24]
DMSJED1188E	Filemode <i>mode</i> is not associated with a directory [RC = 74]
DMSJED1223E	There is no default file pool currently defined [RC = 40]
DMSJLK002E	File <i>fn ft fm</i> <i>dirname</i> not found [RC = 28]
DMSJLK1160E	Directory <i>dirname</i> already open. [RC = 70]
DMSJLK1163E	The CREATE LOCK command failed for <i>fn ft fm</i> <i>dirname</i> [RC = nn]
DMSJLK1184E	Directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSJLK1184E	File <i>fn ft</i> or directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSJLK1184E	File <i>fn ft fm</i> not found or you do not have write authority to it [RC = 28]
DMSJLK1184E	Directory <i>dirname</i> not found or you do not have write authority to it [RC = 28]
DMSJLK1210E	Directory <i>dirname</i> not found [RC = 28]

CREATE LOCK

- DMSJLK1212E You need to be enrolled as a user in file pool *filepoolid* to create a lock on a file or directory [RC=76]
- DMSJLK1214E You have already created a lock of type {EXCLUSIVE|SHARE|UPDATE} on file *fn ft dirname|fm* [RC=28]
- DMSJLK1214E You have already created a lock of type {EXCLUSIVE|SHARE|UPDATE} on directory *dirname* [RC=28]
- DMSJLK1215E A lock of type {EXCLUSIVE|SHARE|UPDATE} on file *fn ft dirname|fm* was already created by another user [RC=28]
- DMSJLK1215E A lock of type {EXCLUSIVE|SHARE|UPDATE} on directory *dirname* was already created by another user [RC=28]
- DMSJLK1291E There are no unused work units available. [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

CREATE NAMEDEF

Use the CREATE NAMEDEF command to create a temporary name (namedef) that can be used by a program instead of:

- file name and file type
- directory name.

Use QUERY NAMEDEF to display namedefs.

Format

CREate NAMedef	$\left\{ \begin{array}{l} fn\ ft \\ dirid \end{array} \right\} \quad namedef \quad [(\text{options...} [])]$ <p>Options: [REPlace]</p>
-----------------------	---

Operands

fn ft

specifies the file name and file type of the file that the namedef represents.

dirid

specifies the directory name that the namedef represents. For a detailed description of *dirid*, see "Naming Shared File System (SFS) Directories" on page 4.

namedef

specifies a 1 to 16 character temporary name to refer to a file name and file type or a directory name.

Option

REPlace

means that if the namedef already exists to replace it with the namedef you are creating.

Usage Notes

1. You may use up to 16 characters for the namedef; however, the first character must be an alphabetic character. Remaining characters may be alphabetic or numeric.
2. The namedef that you create stays in effect until your CMS session ends or until another CREATE NAMEDEF is issued for the same namedef with the REPLACE option. You may delete namedefs by using the DELETE NAMEDEF command.
3. If any program abends occur, all the namedefs are deleted. For example, if you issue the Immediate command, HX, while a program is running, all the namedefs are deleted.
4. If a file or directory is renamed, the namedef continues to refer to the original name of the *fn ft* or *dirid*, both of which may no longer exist.
5. If the directory name is specified as a file mode, for example,

CREATE NAMEDEF

```
create namedef +a.proj1 dirname
```

the namedef DIRNAME refers to the subdirectory PROJ1 of the directory accessed as A. If you access another directory as A, the namedef continues to refer to the original subdirectory.

6. If you are not sure which namedefs have already been created, use the QUERY NAMEDEF command.

Example

Consider a program in which the file identifier FILEID DIRNM is coded in the parameter list for an OPEN function. By using two namedefs, you can have the program process different files and directories without changing the code and recompiling the program. For example, issue:

```
create namedef data1 data fileid  
create namedef poola:john.870726 dirnm
```

Then run the program. The program would OPEN the DATA1 DATA file in the POOLA:JOHN.870726 directory. To run the program with another file simply change the namedef:

```
create namedef data2 data fileid (rep
```

Then run the program. Since the namedefs are resolved at run time you don't need to have them defined prior to compiling your program.

Messages and Return Codes

DMSJCR065E	REPLACE option specified twice [RC = 24]
DMSJED069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSJED109S	Virtual storage capacity exceeded [RC = 104]
DMSJED1187E	Too many subdirectory levels in <i>dirid</i> [RC = 24]
DMSJED1188E	Filemode <i>mode</i> is not associated with a directory [RC = 74]
DMSJED1189E	Filemode <i>mode</i> is associated with a top directory [RC = 24]
DMSJED1223E	There is no default file pool currently defined [RC = 40]
DMSJNA1191E	Namedef <i>namedef</i> already exists [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813

CSLLIST

Use the CSLLIST command to display the contents of a callable services library (CSL).

Format

CSLList	<i>libname</i> [(options... [])] Options: [IN <i>fm</i> IN <i>dirid</i> SEGment <i>segid</i>] [Append] [PROFile <i>fn</i>]
----------------	---

Operands

libname

specifies the callable services library to be listed. If neither the IN nor SEGMENT option is specified, saved segments are searched first, and then files are searched using the CMS file search order.

Options

IN *fm*

identifies the file mode of the accessed minidisk or SFS directory containing the library.

IN *dirid*

identifies the SFS directory containing the library. See "Naming Shared File System (SFS) Directories" on page 4 for a description of how to specify *dirid*.

SEGment *segid*

identifies the saved segment containing the library.

Append

specifies that the routines of another library are to be added to those currently being displayed. This option is only valid from within a CSLLIST full-screen session.

PROFile *fn*

specifies that an XEDIT profile named "*fn* XEDIT" should be used when entering the CSLLIST environment. If this option is not used, a macro named PROFCLST XEDIT is invoked.

Usage Notes

1. You can use the special command EXECUTE from the CSLLIST screen. The EXECUTE command allows you to issue commands that use the routines displayed by CSLLIST. See "EXECUTE" on page 794 for more information.
2. When you invoke CSLLIST you are under control of the System Product Editor (XEDIT). At this time, you are editing the following CMS file:

fn = user ID you are logged on with
ft = CSLLIST

fm = the first available R/W disk or directory, or S disk if no R/W disk or directory is available.

Each line of this file contains:

- a command area
- a routine name
- a keyword showing where the routine resides (DASD for disk or directory; SEGMENT for logical saved segment)
- the library name
- the logical segment name or file type
- the file mode or a blank if the library resides in a segment.

3. Entering CMS commands from CSLLIST

Begin CMS commands with "CMS" to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.

4. Tailoring the CSLLIST Command Options

The DEFAULTS command can be used to set up options and/or override command defaults for CSLLIST. The options that you specify from the command line when issuing CSLLIST, override the defaults specified in the DEFAULTS command. This allows you to customize the defaults yet override them when desired.

5. Issuing Commands from the CSLLIST Screen

You can issue commands directly from the line that displays a routine. To do this, move the cursor to the line containing the CSL routine and type the command in the command (Cmd) column, then press ENTER to execute the command. After you issue the command, press the PF2 key to refresh your screen so that an updated screen is displayed to type other commands.

If a command is longer than the command space provided on the screen, just continue typing over the information in the line. You may type over the entire line displayed up to column 79.

To move the cursor to the command line, press the PF12 key (or the ENTER key). Press the PF12 key again to move the cursor back to its previous position on the list.

6. Using Symbols as Part of a Command

Use the following symbols if the command you want to execute uses the CSL routine or library name on the line where the routine is displayed:

/n means the CSL routine name that is displayed on the line.

/l means the library name that is displayed on the line. If the routine resides on a minidisk or in a directory, **/l** means *libname* IN *fm*.

/o means execute the line as is and omit appending anything.

7. Special Symbols Used Alone

The following special symbols can be typed alone on the lines of the CSLLIST display:

= means execute the previous command for this routine. The command is executed starting with the top of the screen. For example, if you issue the RTNDROP /N command on the top line, you can type an equal sign on any other line(s). Those routines preceded by equal signs are dropped when you press ENTER.

? displays the last executed command on the line that the ? was entered.

/ means make this line the current line. The current line for CSLLIST is the first routine on the screen.

8. Default Key Settings

Entering the CSLLIST command executes the PROFCLST XEDIT macro, unless you specify a different macro as an option in the CSLLIST command. If you want to always use another profile, see the DEFAULTS command. The keys are set to the following values by PROFCLST XEDIT:

Key	Setting	Action
ENTER	Execute	Execute command(s) typed on routine line(s) or on the command line.
PF 1	Help	Displays CSLLIST command description.
PF 2	Refresh	Update the list to indicate new routines, erased routines, etc., using the same parameters as those specified when CSLLIST was invoked.
PF 3	Quit	Exit from CSLLIST.
PF 4	Template	Displays template information for that routine.
PF 5	Sort (routine)	Sort alphabetically by routine name.
PF 6	Rtnmap	Lists the CSL routines that are currently loaded.
PF 7	Backward	Scroll back one screen.
PF 8	Forward	Scroll forward one screen.
PF 9	Rtnload	Loads the routine at the cursor by issuing the command RTNLOAD /n (FROM /l if the routine is in a saved segment. If the routine at the cursor is in an accessed minidisk or directory, the command RTNLOAD /n (FROM /l IN <i>fm</i> is issued.
PF 10	Rtndrop	Drops the routine at the cursor by issuing the command RTNDROP /n at the cursor.
PF 11	Sort (libname)	Sorts the list alphabetically by library name and then by routine name.
PF 12	Cursor	If the cursor is in the list area, move it to the command line. If the cursor is on the command line, move it back to its previous location in the list (or to the current line).

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFCLST XEDIT macro sets synonyms that you can use to sort your CSLLIST display. The synonyms are:

SNAME Sorts the list alphabetically by routine name.

SLIB Sorts the list alphabetically by library name.

9. The following commands are related to CSLLIST:

- RTNLOAD - loads a CSL routine
- RTNMAP - lists the CSL routines that are currently loaded
- RTNDROP - drops a loaded CSL routine

10. If you specify a directory name in the IN *dirid* option and the directory is not accessed, the directory is temporarily accessed as the next available file mode letter. When you exit CSLLIST the file mode is released.

Examples

1. Suppose someone with the user ID JOHNDOE has a callable services library named COURSES on his disk or directory accessed as A. The library contains five routines that each generate a description for a class. If this user enters the command

```
csllist courses (in a
```

the following will be displayed:

```
JOHNDOE CSLLIST A0 V 130 Trunc=130 Size=5 Line=1 Col=1 Alt=0
Cmd  Routine      Resident  Libname   Segid/Type  Mode
-
      GEOGRAPH    DASD     COURSES   CSLLIB      A1
      HISTORY     DASD     COURSES   CSLLIB      A1
      PHYSICS     DASD     COURSES   CSLLIB      A1
      PSYCH       DASD     COURSES   CSLLIB      A1
      CALCULUS    DASD     COURSES   CSLLIB      A1

1= Help    2= Refresh 3= Quit   4= Template 5= Sort(routine) 6= Rtnmap
7= Backward 8= Forward 9= Rtnload 10= Rtdrop 11= Sort(libname) 12= Cursor
====>
```

2. Suppose JOHNDOE also has a callable services library named TRIG on a logical saved segment called JDSEG1. The library contains three routines that calculate trigonometry formulas. If this user enters the command

```
csllist trig (segment jdseg1
```

the following will be displayed:

```

JOHNDOE CSLLIST A0 V 130 Trunc=130 Size=3 Line=1 Col=1 Alt=0
Cmd  Routine      Resident  Libname   Segid/Type  Mode
-    SINE           SEGMENT  TRIG      JDSEG1
    COSINE        SEGMENT  TRIG      JDSEG1
    TANGENT       SEGMENT  TRIG      JDSEG1

1= Help    2= Refresh 3= Quit    4= Template 5= Sort(routine) 6= Rtnmap
7= Backward 8= Forward 9= Rtnload 10= Rtn drop 11= Sort(libname) 12= Cursor
====>

```

3. From within the CSLLIST environment shown above in example 2, JOHNDOE can issue the command

```
csllist courses (in a append
```

to append information about his COURSES library.

```

JOHNDOE CSLLIST A0 F 130 Trunc=130 Size=8 Line=1 Col=1 Alt=1
Cmd  Routine      Resident  Libname   Segid/Type  Mode
-    SINE           SEGMENT  TRIG      JDSEG1
    COSINE        SEGMENT  TRIG      JDSEG1
    TANGENT       SEGMENT  TRIG      JDSEG1
    GEOGRAPH      DASD     COURSES   CSLLIB      A1
    HISTORY       DASD     COURSES   CSLLIB      A1
    PHYSICS       DASD     COURSES   CSLLIB      A1
    PSYCH         DASD     COURSES   CSLLIB      A1
    CALCULUS      DASD     COURSES   CSLLIB      A1

1= Help    2= Refresh 3= Quit    4= Template 5= Sort(routine) 6= Rtnmap
7= Backward 8= Forward 9= Rtnload 10= Rtn drop 11= Sort(libname) 12= Cursor
====>

```

4. When the cursor is positioned in the command area of a listed routine, pressing PF4 displays template information for that routine. (CSL routine templates are described in the *VM/SP Application Development Guide for CMS*.)

For example, positioning the cursor beside the routine SINE above and pressing PF4 would display the following:

```

SINE      $TEMPLAT A0 F 80 Trunc=80 Size=4 Line=1 Col=1 Alt=0
I/O      Datatype      Length      Required      Total
OUTPUT   BIN           4           4           4
IN       BIN           2           4           4
IN       BIN           2           4           4
OUT      BIN           4           4           4

1= Help   2=           3= Return 4=           5=           6=
7= Backward 8= Forward 9=           10=          11=          12= Cursor

====> _

```

5. When the cursor is positioned in the command area of a listed routine, pressing PF4 displays template information for that routine.

This is also under control of the System Product Editor. A new set of PF keys is defined as shown in the example. After viewing this template information, you can enter QUIT on the command line or press the appropriate PF key to return to the CSLLIST environment.

Responses

When a command is executed, one of the following symbols is displayed in the "Cmd" space to the left of the routine for which the command was executed.

- * Means the command was executed successfully (RC = 0).
- *n Is the return code from the command executed (RC = n).
- *? Means that the command was an unknown CP/CMS command (RC = -3).

The following response can also appear directly on the CSLLIST screen:

- * Library libname not found

Messages and Return Codes

- DMSCRI639E Error in *rtname* routine; return code was *retcode*
- DMSCRI1097E Routine *rtname* not found [RC = 28]
- DMSCRI1136E Unable to gain access to library *libname* [RC = 28]
- DMSWCL065E *option* option specified twice [RC = 24]
- DMSWCL066E *opt1* and *opt2* are conflicting options [RC = 24]
- DMSWCL651E APPEND must be issued from CSLLIST [RC = 40]
- DMSWCL1100E No filemode is available to access *dirname* [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

DEBUG

Use the DEBUG command to display status information following ABEND processing.

Format

DEBUG	
-------	--

The DEBUG command has no operands.

Usage Notes

1. CMS does not support the DEBUG subcommand environment. The functions provided by the DEBUG subcommands are still available using CP DISPLAY, TRACE and PER.
2. In CMS, program and external interrupts no longer call DEBUG.
3. The DEBUG command is valid only if you enter it from the VM READ of CMS ABEND processing. (This is the VM READ that is typically produced by the HX immediate command or by an application program that abnormally terminates). If you enter the DEBUG command from outside the VM READ of ABEND processing, CMS returns an error message.
4. When you issue the DEBUG command, CMS displays information on your display and then returns you to the VM READ of CMS ABEND processing. You may then reenter the DEBUG command or type any CMS command to exit ABEND processing.

Response

The following sample shows the information that the DEBUG command displays:

```

DMSDBG989I The state of the virtual machine at time of
          ABEND follows:
Mode of virtual machine = XA   PSW type = ECMODE
PSW = 00C02000 82FCDE72
GR  0 = FFFFFFFF 0000D800 00000008 0000D808
GR  4 = 82818495 4085A6A2 000000FF 00000000
GR  8 = E2859584 40948540 A3964083 8193864B
GR 12 = 02FCDC80 0000D700 000329FC 00000010
FPR 0 = 0000D28595FF0700      .27742033723982779 E-79
FPR 2 = E000000000000010      -.75557863725914323 E 23
FPR 4 = 0000000000000000      .00000000000000000 E 00
FPR 6 = 4080000000000000      .50000000000000000 E 00
External old PSW = FF002000 00D0DF70
SVC old PSW = FF802000 82FCDEE8
Program old PSW = 00020000 000890D0
Machine-check old PSW = 00020000 60736530
Input/Output old PSW = FF802000 82FCDD8C
Vector status register= 00001200

```

Note: If the Vector Facility is not in use or is not installed, the display will end with the Input/Output old PSW.

Messages and Return Codes

- DMSDBG906E DEBUG command not allowed at this time.
- DMSDBG989I The state of the virtual machine at time of ABEND follows:
- DMSDBG998E Error saving the Vector Status Register.

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DEFAULTS

Use the DEFAULTS command to set up default options, or display the current default options for the commands shown in the table below.

Each time you enter one of these commands, the options specified in the DEFAULTS command are in effect. However, the options specified with each invocation of the various commands override the ones set up in the DEFAULTS command. Thus, you can customize the options by using DEFAULTS, yet override them when you desire.

Format

DEFAULTS	$\left[\begin{array}{l} \text{Set } \textit{command options...} \\ \text{List } [\textit{command}] \end{array} \right]$
-----------------	--

Operands

Set

specifies that default options are to be set up for the command indicated.

List

specifies that the current default options for the command indicated are to be displayed. If no command is specified, all the commands listed below and the current default options are displayed.

command

is one of the commands listed below.

options

is one or more options associated with a particular command, as shown below.

The commands and options that can be specified as defaults are listed below. Valid abbreviations for both the command names and the keyword options are indicated by uppercase letters. Mutually exclusive options are listed one under the other. The system defaults are underscored.

Command Name	Options
CMSSERV	CUT <u>DFT</u>
CSLList	Profile fn <u>Profile</u> <u>PROFCLST</u>
Dirlist	Profile fn <u>ALL</u> Dirlist fn <u>Profile</u> <u>PROFDLST</u> ACCessed <u>NODirlist</u>
DISK Load	<u>NOReplace</u> <u>MINPrompt</u> Replace <u>NOPrompt</u> <u>FULLPrompt</u>

2. The FILELIST command has three profiles which set up the three FILELIST screens. By default, PROFILE PROFFLST is used for the STATS screen, PROFILE2 PROFFSHR is used for the SHARE screen, and PROFILE3 PROFFSEA is used for the SEARCH screen.

You can use the DEFAULTS command to change these three profiles. For example, if you want to use a profile named MYPROF XEDIT instead of PROFFSHR XEDIT for the SHARE screen, and you want to leave the other two profiles as the default system profiles, you would enter:

```
defaults set filelist profile2 myprof
```

3. You must be an authorized class B user to send messages with the CP MSGNOH command.
4. If you want to issue DEFAULTS from an EXEC program, you should precede it with the EXEC command; that is, specify

```
exec defaults
```

Example

To change the SENDFILE command default from NEW to OLD, you would enter:

```
defaults set sendfile old
```

Responses

The following is a list of your default options for the cmdname command:

```
option...
```

```
.  
.  
.
```

To change these default options enter 'DEFAULTS Set Cmdname Opt1 <Opt2..>'.
.

The following default options have been set:

```
commandname option...
```

To change any default options enter 'DEFAULTS Set Cmdname Opt1 <Opt2..>'.
.

Messages and Return Codes

DMSWDF003E Invalid option: *option* [RC=24]

DMSWDF641E No options specified [RC=24]

DMSWDF653E Error executing GLOBALV, RC=*nn* [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DELETE LOCK

DELETE LOCK

Use the DELETE LOCK command to release an explicit lock on a Shared File System (SFS) directory or a file in an SFS directory. To remove a lock, you must be the creator of the lock and have read or write authority to the file or directory. Explicit locks are created using the CREATE LOCK command.

Format

DELeTe LOCK	$\left[\begin{array}{cc} fn & ft \\ * & * \end{array} \right] \text{ dirid } [(\text{options...} [])]$
	<p><u>Options:</u></p> $\left[\begin{array}{l} \text{TYPE} \\ \text{NOType} \\ \text{STACK } \left[\begin{array}{l} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{LIFO} \\ \text{FIFO} \end{array} \right]$

Operands

fn ft

specifies the name of the file that is to be unlocked. Special characters (* and %) can be used to designate a set of files. See "Pattern Matching" on page 8 for more information on these special characters.

dirid

specifies the directory name from which you wish to release the lock. If *fn* and *ft* are specified, this is the directory which contains the file that is to be unlocked. If *fn* and *ft* are not specified, this is the name of the directory which is to be unlocked. If further detail is needed on *dirid*, see "Naming Shared File System (SFS) Directories" on page 4.

Options

TYPE

displays at the terminal the names of the files or the SFS directory for which a lock has been deleted.

NOType

suppresses the display of information at the terminal. NOType is the default.

STACK [FIFO]

STACK LIFO

places the information in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

Usage Notes

1. If special characters are used for *fn* or *ft*, locks are not deleted in the directory for:
 - a. subdirectories
 - b. erased or revoked aliases
 - c. files that you are not authorized to read or write.

Processing continues for the remaining file names that match the specified pattern.

2. If special characters are used to delete locks on a set of files, the directory containing the files must be closed.
3. You can delete a lock on a file even if you have the file open, as long as the DELETE LOCK command is issued on a different work unit.
4. If the DELETE LOCK command is issued from an exec, on a work unit that has not been committed the command will fail.
5. You can invoke the DELETE LOCK command from the command line, from an exec, or as a function from a program. No error messages are issued if DELETE LOCK is invoked:
 - As a function from a program
 - From a CMS exec file that has the &CONTROL NOMSG option in effect
 - From an EXEC2 exec where CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect

Messages and Return Codes

DMSDEL065E	FROM option specified twice [RC = 24]
DMSDEL066E	<i>option1</i> and <i>option2</i> are conflicting options [RC = 24]
DMSDEL1132E	Invalid number of operands [RC = 24]
DMSJED069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSJED109S	Virtual storage capacity exceeded [RC = 104]
DMSJED1187E	Too many subdirectory levels in <i>dirname</i> [RC = 24]
DMSJED1188E	Filemode <i>mode</i> is not associated with a directory [RC = 74]
DMSJED1189E	Filemode <i>mode</i> is associated with a top directory [RC = 24]
DMSJED1223E	There is no default file pool currently defined [RC = 40]
DMSJLK002E	File <i>fn ft fm dirname</i> not found [RC = 28]
DMSJLK1139E	You are not authorized to issue this command [RC = 76]
DMSJLK1160E	Directory <i>dirname</i> already open. [RC = 70]
DMSJLK1163E	The DELETE LOCK command failed for <i>fn ft fm dirname</i> [RC = <i>nn</i>]
DMSJLK1184E	Directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSJLK1184E	File <i>fn ft</i> or directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSJLK1206W	There are no locks for <i>fn ft fm dirname</i> [RC = 4]
DMSJLK1209E	Nickname <i>nickname</i> resolved to more than one userid; lock(s) can be deleted from only one userid at a time [RC = 88]

DELETE LOCK

DMSJLK1210E Directory *dirname* not found [RC=28]
DMSJLK1291E There are no unused work units available. [RC=88]
DMSJNL637E Missing nodeid for the AT operand [RC=24]
DMSJNL647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
DMSJNL647E Localid not specified for *userid* at *node* in *userid* NAMES file
[RC=32]
DMSJNL653E Error executing *command rc=nn* [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

DELETE NAMEDEF

Use the DELETE NAMEDEF command to delete a temporary name (namedef).

Format

DELeTe NAMeDef	$\left\{ \begin{array}{c} \textit{namedef} \\ * \end{array} \right\}$
-----------------------	---

Operands

namedef

specifies a 1 to 16 character temporary name previously created by the CREATE NAMEDEF command. If an asterisk (*) is specified, all current namedefs are deleted.

Usage Notes

1. You can invoke the DELETE NAMEDEF command from the command line, from an exec, or as a function from a program. No error messages are issued if DELETE NAMEDEF is invoked:
 - As a function from a program
 - From a CMS exec file that has the &CONTROL NOMSG option in effect
 - From an EXEC2 exec where CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect

Messages and Return Codes

DMSJNA1147E Storage management error trying to free storage [RC = 104]

DMSJNA1192E Namedef *namedef* not found [RC = 28]

DMSJNA1193E There are no namedefs to be deleted [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DESBUF

Use the DESBUF command to clear the console and program stack input and output buffers.

Format

DESBUF	
--------	--

Usage Notes

1. Note that DESBUF clears the output buffers as well as the input buffers. Use the CONWAIT command before DESBUF to halt program execution until all output lines are displayed at the terminal.

Warning: Be careful when using the DESBUF command because the input and output console, and program stack buffers are used to communicate information between programs.

DIRLIST

Use the DIRLIST command to display a list of Shared File System (SFS) directories for a specified directory structure.

The LISTDIR and DIRLIST commands display identical information, but in the DIRLIST environment, information is displayed under the control of the System Product Editor. You can issue XEDIT subcommands to manipulate the list itself. You can also issue CMS commands against the directories directly from the displayed list.

Format

DIRList	[<i>dirid</i>] [(options...[])]
	Options: [ACC essed] [PROFile <i>fn</i>] [<u>ALL</u>] [APP end] [DIRlist <i>fn</i>] [NODirlist]

Operands

dirid

specifies the directory where the list begins. The list contains this directory and all directories below it in the directory structure. Only the directories for which you have read or write authority are listed. If *dirid* is not specified, directories are listed beginning with your top directory. (See the ACCESSED option for an exception to this rule).

Options

ACCessed

lists only the accessed directories in the specified directory structure. If *dirid* is not specified with this option, all accessed directories in the CMS search order are listed.

ALL

lists all the directories in the specified directory structure for which you have read or write authority, whether they are accessed or not. ALL is the default.

APPend

lists the directories at the end of the existing list. This option is only valid when issued from within the DIRLIST environment.

PROFile *fn*

specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the DIRLIST command. If PROFILE *fn* is not specified, a macro named PROFDLST XEDIT is invoked. For more information, see Usage Note 9 under "Default Key Settings."

DIRLIST

DIRlist *fn*

specifies that a file called *fn* DIRLIST already contains a list of directories produced by an earlier invocation of DIRLIST. This existing list of directories is displayed. If you have also specified *dirid* with this option, it is ignored. For further information, see Usage Note 4 under "Saving a List of Directories."

NODirlist

specifies that no list of directories is to be used. NODIRLIST is the default.

Usage Notes

1. Format of the List

When you invoke the DIRLIST command you are placed in the XEDIT environment editing a file named "*userid* DIRLIST A0." Each line in this file contains:

- a command area
- a file mode (a dash is displayed for file mode if directory is not accessed).
- the directory name (up to 70 characters are displayed on a line). If the directory name is too long for the line, you can scroll right to see the remainder of the name.

The full power of XEDIT is available while commands are issued against the list. For example, you can scroll through the list of directories, locate a particular directory, etc.

However, some XEDIT subcommands are inappropriate in this environment. Subcommands that alter the format or the contents of "*userid* DIRLIST" (for example, SET LRECL, SET TRUNC, SET FTYPE, or SET LINEND) may cause unpredictable results.

2. Entering CMS commands from DIRLIST

Begin CMS commands with "CMS" to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.

3. Using special commands from DIRLIST: EXECUTE, DISCARD, and AUTHLIST

The EXECUTE command allows you to issue commands that use the directories displayed by DIRLIST. The DISCARD command allows you to erase the directories displayed by DIRLIST. The AUTHLIST command displays authority information about the directories displayed by DIRLIST. For more information, see *Chapter 4 - Special Commands Used in Command Environments*.

4. Saving a List of Directories

To save a list of displayed directories, issue FILE or SAVE from the command line. The list is saved in a file named "*userid* DIRLIST" until the next time you issue FILE or SAVE on the list.

To save a particular list of directories, file it under a different file name. You have two choices on how to do this:

- Issue FILE from the command line and specify a different *filename*. For example, you could issue FILE MYDIRS.
- Issue FILE from the command line and then use the RENAME command.

Saving a list of directories is useful when you want certain directories to be listed each time you issue DIRLIST. For example, to combine several users' directories together in one list, you can use the APPEND option from DIRLIST. After saving the list, you may now issue DIRLIST with the DIRLIST option to display all the directories previously saved.

5. Tailoring the DIRLIST Command Options

The DEFAULTS command can be used to set up options and/or override command defaults for DIRLIST. The options that you specify from the command line when issuing DIRLIST will override the defaults specified in the DEFAULTS command. This allows you to customize the defaults yet override them when desired. For example, when you have set DIRLIST as your default option using the DEFAULTS command, CMS searches for a file called "*userid* DIRLIST" to use for the list of directories. Refer to the DEFAULTS command description for more information.

6. Issuing Commands from the DIRLIST screen

You can issue commands directly from the line that displays a directory. To do this, move the cursor to the line containing the directory and type the command in the command (Cmd) column, then press ENTER to execute the command. After you issue the command, press the PF2 key to refresh your screen so that an updated screen is displayed to type other commands.

If a command is longer than the command space provided on the screen, just continue typing over the information in the line. You may type over the entire line displayed up to column 79.

To move the cursor to the command line, press the PF12 key (or the ENTER key). Press the PF12 key again to move the cursor back to its previous position on the list.

When you enter a command without any symbols, the directory name is automatically appended to the end of the command.

7. Using Symbols as Part of a Command

Use the following symbols if the command you want to execute has operands or options that follow the directory name on the line where the directory is displayed:

/ is equivalent to the directory name.

/m means the file mode that is displayed on the line.

/d means the directory name that is displayed on the line.

/o means execute the line as is and omit appending anything.

See the FILELIST command for some examples of how to use these symbols.

8. Using Special Symbols

The following special symbols can be typed alone on the lines of the DIRLIST display:

= means execute the previous command for this directory. Commands are executed starting with the top of the screen. For example, if you issue the ERASE command on the top line, you can type an equal sign on any other line(s). Those directories preceded by equal signs are erased when you press ENTER.

? displays the last executed command on the line that the ? was entered.

DIRLIST

/ means make this line the current line. The current line for DIRLIST is the first directory on the screen.

9. Setting Defaults for Keys

The PROFDLST XEDIT macro is executed when DIRLIST is issued unless you specify a different macro as an option.

The keys are set to:

Key	Setting	Action
ENTER	Execute	Executes the command(s) typed on the directory line(s) or on the command line.
PF 1	Help	Displays a help screen that gives the DIRLIST command description.
PF 2	Refresh	Updates the list to indicate new directories, renamed directories, etc., using the same operands and options as when DIRLIST was invoked.
PF 3	Quit	Exits from DIRLIST.
PF 4	Sort(fm)	Sorts the list alphabetically by file mode. The SMODE XEDIT macro is used to do this.
PF 5	Sort(dir)	Sorts the list alphabetically by directory name. The SDIR XEDIT macro is used to do this.
PF 6	Auth	Issues an AUTHLIST command for the directory where the cursor is placed. The AUTHLIST information is placed in an XEDIT file named "userid AUTHLIST A0." This is now a new full screen environment with the PF keys set to new values. The AUTHLIST screen indicates which authority you have on the designated directory and the authorities you gave to other users. See "AUTHLIST" on page 789 for more information.
PF 7	Backward	Scrolls backward one screen.
PF 8	Forward	Scrolls forward one screen.
PF 9		Not assigned.
PF10		Not assigned.
PF11	Filelist	Issues the command FILELIST * * for the directory where the cursor is placed. Displays a list of files for the designated directory whether it is accessed or not. If the directory is not accessed, a temporary access is done using the last available file mode in the search order. The file mode is automatically released when you exit this FILELIST screen.
PF12	Cursor	Moves the cursor from its location to the command line or vice versa.

Note: If you have 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 - 12.

In addition to these PF key settings, the PROFDLST XEDIT macro sets the following synonyms that can be used to sort the listed directories:

SMODE Sorts the list alphabetically by file mode. Directories that are not accessed are listed last, sorted alphabetically by directory name. SMODE is an XEDIT macro.

SDIR Sorts the list alphabetically by directory name.

10. Displaying your authorized directories via an XEDIT macro

The following is a sample XEDIT macro that you may wish to duplicate for a convenient way of displaying directories.

```

/*****/
/* GETDIRS XEDIT: */
/* */
/* Issue this XEDIT macro from DIRLIST. It will display all of the */
/* directories that you are authorized for on your current default */
/* file pool. */
/*****/

'TOP'
'QUERY ENROLL USER FOR ALL (FIFO'
If rc = 0 Then Exit
pull .
i = queued()
DO i
  Pull userid .
  Address Command 'LISTDIR' userid||'. (XEDIT'
End
'SDIR'

```

Example

User SMITH enters,

```
dirlist
```

from the command line, and the following screen is displayed:

```

SMITH DIRLIST A0 V 319 Trunc=319 Size=8 Line=1 Col=1 Alt=0
Cmd  Fm Directory Name
A  FPOOL1:SMITH.
-  FPOOL1:SMITH.ADDRESSES
-  FPOOL1:SMITH.ADDRESSES.EMPLOYEES
-  FPOOL1:SMITH.ADDRESSES.MANAGERS
B  FPOOL1:SMITH.ANNIVERSARIES
-  FPOOL1:SMITH.ANNIVERSARIES.EMPLOYEES
-  FPOOL1:SMITH.BIRTHDAYS
-  FPOOL1:SMITH.BIRTHDAYS.EMPLOYEES

1= Help    2= Refresh 3= Quit    4= Sort(fm)  5= Sort(dir) 6= Auth
7= Backward 8= Forward 9=         10=          11= Filelist 12= Cursor

====>
XEDIT 1 File

```

Figure 3. Sample DIRLIST Screen

DIRLIST

See "Responses" on page 106 for an explanation of the information listed in each column.

Responses

Issuing the DIRLIST command with no operands or options displays the following information:

```
Cmd Fm Directory Name
    fm directory name
    .
    .
```

where

Cmd is a command column for typing and entering commands for the directory displayed on the line. You may use special characters such as / and = to represent what is already displayed on the line.

Fm is the file mode letter of the accessed directory

- Dash means the directory is *not* accessed

Directory Name is the complete directory name

When a command is executed on the line where a directory is listed, one of the following symbols is displayed in the Cmd column.

- * means the command was executed successfully (RC = 0).
- *n means the number for the return code (RC = n).
- *? means the command was unknown to CP or CMS (RC = -3).
- *! means the command is not valid in CMS subset. For a list of commands valid in CMS subset mode, see the *CMS User's Guide*.

Messages and Return Codes

DMSJED069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSJED109S	Virtual storage capacity exceeded [RC = 104]
DMSJED1187E	Too many subdirectory levels in <i>dirname</i> [RC = 24]
DMSJED1188E	Filemode <i>mode</i> is not associated with a directory [RC = 74]
DMSJED1189E	Filemode <i>mode</i> is associated with a top directory [RC = 24]
DMSJED1223E	There is no default file pool currently defined [RC = 40]
DMSJLD1160E	Directory <i>dirname</i> already open. [RC = 70]
DMSJLD1184E	Directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSJLD1210E	Directory <i>dirname</i> not found [RC = 28]
DMSOUT105S	Error <i>nn</i> writing file to XEDIT [RC = 100]
DMSSTT002E	File <i>fn ft fm</i> not found [RC = 28]
DMSWDL557E	No more storage to insert lines [RC = 4]
DMSWDL651E	APPEND SMODE must be issued from DIRLIST [RC = 40]
DMSWDL653E	Error executing LISTDIR, rc = <i>nn</i> [RC = <i>nn</i>]
DMSWFL651E	X\$NDIR\$X must be issued from FILELIST or DIRLIST [RC = 40]
DMSWFL1227E	No filemode is available to access directory [RC = 00]
DMSWFL1228E	Error executing access for directory, rc = <i>nn</i> [RC = 00]
DMSWFL1229E	Directory is empty [RC = 00]
DMSWFL1234E	Error executing FILELIST, rc = <i>nn</i> [RC = <i>nn</i>]
DMSWFL1249I	Directory has been temporarily accessed as filemode <i>fm</i> [RC = 00]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

DISK

Use the DISK command to:

- Punch CMS files to the virtual spooled card punch in a special format which allows the punched deck to be restored to a disk or directory in the form of the original file.
- Restore punched decks created by the DISK DUMP command to a file.

Format

DISK	$\left\{ \begin{array}{l} \text{DUMP} \quad fn \ ft \ [fm] \\ \text{LOAD} \quad [(\text{options... } [])] \end{array} \right\}$ <p><u>Options:</u> $\left[\begin{array}{l} \text{Fullprompt} \\ \text{Minprompt} \\ \text{NOPrompt} \end{array} \right]$ $\left[\begin{array}{l} \text{Replace} \\ \text{NOReplace} \end{array} \right]$ $\left[\text{OLDDate} \right]$</p>
------	---

Operands

DUMP *fn ft fm*

punches the specified file (fn ft fm). The file may have either fixed- or variable-length records. After all data is punched, an end-of-file card is created with an N in column 5. This card contains directory information and must remain in the deck. The original file is retained.

LOAD

loads a file or files from the spooled card reader and writes them as CMS files on your disk or directory accessed as A. A file may also be loaded to a disk other than your disk or directory accessed as A through the use of the prompting facility. The file name and file type are obtained from the card stream. If a file exists with the same file name and file type as one of those in the card stream, it is replaced if the REPLACE option is in effect.

The card-image sequence numbers on all files being loaded are checked. A message notifies the user of any record numbers missing or out of order. The file is loaded whether or not a problem is found in the sequence number check.

The DISK LOAD function checks for invalid characters in the file ID field of the reader file to be loaded. If an invalid character is found, message DMSDSK496S is printed at the console informing the user that an invalid file ID has been found in the input record. The file is left in the reader. A file is not loaded when the last card of the reader file does not match the file name, file type, and file mode of the first card in the reader file.

Options for DISK LOAD

Fullprompt

specifies that a prompt is issued for each file in the spool file.

Minprompt

specifies that a prompt is issued when the name of the first (or only) file differs from the name of the spool file; the prompt for the first file is suppressed when it has the same name as the spool file. A prompt is always issued for the second and subsequent files. MINPROMPT is the default.

NOPrompt

specifies that a prompt is not issued to you as a file is received.

Replace

specifies that if a file of the same file name and file type exists on the disk or directory onto which the incoming file is to be loaded, it is to be replaced with this one.

NOReplace

specifies that a file is not received that would overlay an existing file on the receiving disk or directory. NOREPLACE is the default.

OLDDate

when specified, OLDDATE retains the date and time of the most recent update of the file prior to it being sent to your virtual reader. This date becomes the creation date for the file being loaded. Otherwise, the date and time of execution of the DISK LOAD command will be used as the creation date for the output file produced by the DISK LOAD.

Usage Notes

1. To read files with the DISK LOAD command, they must have been created by the DISK DUMP command. To identify the proper method to use in loading spooled reader files, use the 'RDR' command. Also see the 'RECEIVE' command.
2. To load reader files created by DISK DUMP, you must issue the DISK LOAD command for each spool file. For example, if you enter:

```
disk dump source1 assemble
disk dump source2 assemble
```

the virtual machine that receives the files must issue the DISK LOAD command twice to read the files onto the disk or directory. If you use the CP SPOOL command to spool continuous, for example:

```
cp spool punch cont
disk dump source1 assemble
disk dump source2 assemble
cp spool punch nocont close
```

then you only need to issue the DISK LOAD command once to read both files.

You may send multiple files by continuous spooling (using CP SPOOL PUNCH CONT) or by a series of DISK DUMP commands but those methods are discouraged. As a sender you are encouraged to do the following:

- Always use SENDFILE, which resets any continuous spooling options in effect.
- Do not spool the punch continuous.

3. You cannot receive multiple files as one file by spooling your reader continuous (CONT). The DISK LOAD command resets the continuous spooling option and spools your reader NOCONT.

4. Tailoring the DISK Command Options

You can use the DEFAULTS command to set up options and/or override command defaults for DISK. However, the options you specify in the command line when entering the DISK command override those specified in the DEFAULTS command. This allows you to customize the defaults of the DISK command, yet override them when you desire. Refer to the DEFAULTS command description for more information.

5. DISK LOAD loads a file from the reader into a temporary work file called "DISK CMSUT1." The existing file with the same name as the one being loaded from the reader is then erased. The name of the temporary work file just created is changed to the name of the work file just read in. If the file you are loading has the name "DISK CMSUT1," it is changed to "DISK CMSUT2." "DISK CMSUT1" is a reserved work file name for the DISK command.
6. DISK LOAD or DISK DUMP may cause a file to occupy one extra block on the disk. If the file is close to filling or exactly fills the last block on a 512, 1024, 2048, or 4096 formatted disk, the last record produced by the DISK DUMP or DISK LOAD may be filled with X'00's causing the file to occupy one extra block consisting of X'00's on the disk.
7. If you specify the FULLPROMPT or MINPROMPT option the valid responses include:
- One of the digits specified in the prompt
 - One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	If this file is one of a set of files that constitutes a single spool file, the file is not received and prompting continues for the next file, if there is one. If this is the last file of a set of files or if this is the only file in the spool file, the command is ended.
1 or Yes	Receives the file under the name <i>fn1 ft1 fm1</i> (or <i>fn3 ft3 fm3</i>).
2 or Quit	Ends the command.
3 or Rename	Requests prompt message DMSDSK1080R so that the incoming file can be received using a different name.

8. If you receive prompt message DMSDSK1081R the valid responses include:
- One of the digits specified in the prompt
 - One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	Does not receive the file under the name <i>fn ft fm</i> and repeats the original prompt message DMSDSK1080R. This allows you to specify a different name for the incoming file.

1 or Yes Receives the file under the name *fn ft fm*.

2 or Quit Ends the command.

9. If you encounter any errors when you load a reader spool file, the file remains in the reader and is not purged by the DISK command. This occurs regardless of whether you have spooled the reader HOLD or NOHOLD. This protects you from losing reader spool files when an error is encountered. If the file is empty or unwanted, you can purge the file from your reader.

Responses

1. When DISK LOAD has completed loading each incoming file, you receive one of the following responses, depending on the situation.
 - If the incoming file (*fn1 ft1 fm1*) does not already exist and it is received without being renamed, you receive
fn1 ft1 fm1 created
 - If the incoming file (*fn1 ft1 fm1*) is renamed to a file name (*fn2 ft2 fm2*) that does not already exist, you receive
fn2 ft2 fm2 created from fn1 ft1 fm1
 - If the incoming file (*fn1 ft1 fm1*) is copied to an existing data set that has the same name as the incoming file, you receive
fn1 ft1 fm1 replaced
 - If the incoming file (*fn1 ft1 fm1*) is copied to an existing file (*fn2 ft2 fm2*) with a name different from that of the incoming file, you receive
fn2 ft2 fm2 replaced by fn1 ft1 fm1
 - If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*), but is given a mode (*fm1*) that differs from the mode of the existing file (*fm2*), you receive
fn1 ft1 fm1 replaced fn2 ft2 fm2
 - If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*), but is given a mode (*fm3*) that differs from the mode of the existing file (*fm2*), you receive
fn3 ft3 fm3 replaced fn2 ft2 fm2 sent as fn1 ft1 fm1
2. If you specify the FULLPROMPT or MINPROMPT option, one of these prompts is displayed:

DMSDSK1079R Receive *fn1 ft1 fm1*?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* and replace the existing file
 of the same name?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* and replace *fn2 ft2 fm2*?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3*?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and replace
 the existing file of the same name?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and replace
fn2 ft2 fm2?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

- The file ID *fn1 ft1 fm1* is the name from the card stream in the spool file.
 - The phrase “and replace the existing file of the same name?” appears when the operation replaces an existing file and the file mode of that file is the same as *fm1*.
 - The phrase “and replace *fn2 ft2 fm2*.” appears when the operation replaces an existing file and the file mode of that file is not *fm1*.
 - The file ID *fn3 ft3 fm3* is the name from the card stream in the spool file that you may specify when the name differs from the name of the incoming file.
3. If you respond with a 3 (or RENAME) to prompt message DMSDSK1079R, the following message appears and you must enter a file ID in the form *fn [ft [fm]]*.

DMSDSK1080R Enter the new name for *fn ft fm*

4. If you respond to prompt message DMSDSK1080R with a file ID that names an existing file, you receive this prompt:

DMSDSK1081R Replace *fn ft fm*?
 Reply 0 (NO), 1 (YES), or 2 (QUIT)

Messages and Return Codes

DMSDSK002E	File <i>fn [ft [fm]]</i> not found
DMSDSK014E	Invalid function <i>function</i>
DMSDSK024E	File <i>fn [ft fm]</i> already exists[; specify REPLACE option] [RC = 28]
DMSDSK037E	Filemode <i>mode[(vdev)]</i> is accessed as read/only
DMSDSK047E	No function specified
DMSDSK054E	Incomplete fileid specified
DMSDSK062E	Invalid * in fileid
DMSDSK069E	Filemode <i>mode[(vdev)]</i> not accessed
DMSDSK069E	Output filemode <i>mode[(vdev)]</i> not accessed
DMSDSK070E	Invalid parameter <i>parameter</i>
DMSDSK077E	End card missing from input deck

DMSDSK078E Invalid card in input deck
 DMSDSK078W Sequence error detected loading *fn ft*--expected *seqno1* found
seqno2
 DMSDSK104S Error *nn* reading file *fn ft fm* from disk or directory
 DMSDSK105S Error *nn* writing file *fn ft fm* on disk or directory
 DMSDSK109S Virtual storage capacity exceeded [RC = 104]
 DMSDSK118S Error punching file
 DMSDSK124S Error reading card file
 DMSDSK205W Reader empty or not ready
 DMSDSK257T Internal system error at address *address* (offset *offset*)
 DMSDSK445W Invalid data in sequence field, bypassing sequence check
 DMSDSK496S Invalid fileid *fn ft fm* found in input record [RC = 100]
 DMSDSK550W Date/Time data not present for file *fn ft*
 DMSDSK639E Error in *routine* routine; return code was *nnnn*
 DMSDSK671E Error loading file *fn ft fm*; rc = *nn* from RENAME
 DMSDSK1123E Unknown response *text* ignored
 DMSDSK1124W Spool file *spoolid* has been left in your reader because one or more
 files were not received [RC = 1]
 DMSDSK1138E Filesharing conflict involving file *fn ft fm* [RC = 70]
 DMSDSK1262S Error *nnn* opening file *fn ft fm* [RC = 31|55|70|99|100]
 DMSDSK1262S Error *nnn* closing file *fn ft fm* [RC = 31|100]
 DMSDSK1285S Default option *text* is invalid [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DLBL

Use the DLBL command:

- In CMS/DOS, to define VSE and CMS sequential files for program input/output; to identify VSE files and libraries; to define and identify VSAM catalogs, clusters, and data spaces; and to identify VSAM, VSE, or CMS files used for VSAM program input/output and access method services functions. In many situations, VSE/VSAM does not require the DLBL command. Information on when a DLBL statement is required can be found in the *VSE/VSAM Programmer's Reference*.
- In CMS, to define and identify VSAM catalogs, clusters, and data spaces; to identify VSAM files used for program input/output; and to identify input/output files for AMSERV.

Format

DLBL	<pre> ddname { fm } [CMS fn ft] [(option A option B())] { DUMMY } [CMS FILE ddname] ddname { fm } [DSN qual1 [.qual2...qualn]] { DUMMY } [DSN qual1 [qual2...qualn]] { DUMMY } [DSN ?] [(optionA optionB optionC ())] { ddname } CLEAR * </pre> <p>Option A: [SYS xxx]</p> <p>Option B: [PERM] [CHANGE NOCHANGE]</p> <p>Option C: [VSAM] [EXTENT] [CAT catdd] [BUFSP nnnnn] [MULT]</p>
-------------	--

Note: The operands and options of the DLBL command are described below. Usage notes are provided for general usage, followed by additional notes for CMS/DOS users, and then additional notes for OS VSAM users.

Operands*ddname*

specifies a one- to seven-character program ddname (OS) or file name (VSE), or dname (as specified in the FILE parameter of an access method services control statement). An asterisk (*) entered with the CLEAR operand indicates that all DLBL definitions, except those that are entered with the PERM option, are to be cleared.

fm

specifies a valid CMS file mode letter and optionally, file mode number. A letter must be specified; if a number is not specified, it defaults to 1. The disk or directory must be accessed when the DLBL command is issued. In the CMS/DOS and CMS/VSAM environments, file modes "R" and "T" cannot be used on the DLBL command. This is because "R" and "T" are used as abbreviations for reader and terminal in the CMS/DOS and CMS/VSAM environments.

DUMMY

specifies that no real I/O is to be performed. A read operation results in an end-of-file condition and a write operation results in a successful return code. DUMMY should not be used for OS VSAM data sets (see Usage Note 3).

CLEAR

removes any existing definitions for the specified ddname. Clearing a ddname before defining it ensures that a file definition does not exist and that any options previously defined with that ddname no longer have any effect.

CMS *fn ft*

indicates that this is a CMS file, and the file identifier (*fn ft*) that follows is a CMS file name and file type.

FILE *ddname* is the default CMS file identifier associated with all non-CMS data sets. (See Usage Note 3 for CMS/DOS users.)

DSN

indicates that this is a non-CMS file.

?

indicates that you are going to enter the data set name interactively. When prompted, you enter the data set name or file ID in its exact form, including embedded blanks, hyphens, or periods.

qual1 [.qual2...qualn]

-- or --

qual1 [qual2...qualn]

is an OS data set name or VSE file ID. Only data sets named according to standard OS conventions may be entered this way; you may omit the periods between qualifiers, or specify the full dataset name, including periods between qualifiers. (See Usage Note 2.)

Options**SYSxxx**

(CMS/DOS only) indicates the system or programmer logical unit that is associated with the disk or directory on which the file resides. The logical unit must have been previously assigned with the ASSGN command. In many situations VSE/VSAM does not require a SYSxxx operand. Thus no previous ASSGN is required. See *VSE/VSAM Programmer's Reference* for information on when the SYSxxx operand is required.

PERM

indicates that this DLBL definition can be cleared only with an explicit CLEAR request. It will not be cleared when the DLBL * CLEAR command line is entered.

All DLBL definitions, including those entered with the PERM option, are cleared as a result of a program abend or HX (halt execution) Immediate command.

CHANGE

indicates that any existing DLBL for this ddname is not to be canceled, but that conflicting options are to be overridden and new options merged into the old definition. Both the ddname and the file identifier must be the same in order for the definitions to be merged. CHANGE is the default.

NOCHANGE

does not alter any existing DLBL definition for the specified ddname, but creates a definition if none existed.

VSAM

indicates that the file is a VSAM data set. This option must be specified for VSAM functions unless the EXTENT, MULT, CAT, or BUFSP options are entered or the ddnames IJSYSCT or IJSYSUC are used.

EXTENT

indicates that you are going to use access method services to define a VSAM catalog, data space, or unique cluster and you want to enter extent information.

MULT

indicates that you are going to reference an existing multivolume data set and you want to enter the volume specifications.

Note: In many situations VSE/VSAM does not require EXTENT or MULT information. See *VSE/VSAM Programmer's Reference* for information on when EXTENT or MULT information is required.

CAT catdd

identifies the VSAM catalog (defined by a previous DLBL definition) which contains the entry for this data set. You must use the CAT option when the VSAM data set you are creating or identifying is not cataloged in the current job catalog. The variable *catdd* is the ddname in the DLBL definition for the catalog.

BUFSP nnnnnn

specifies the number of bytes (in decimal) to be used for I/O buffers by VSAM data management during program execution, overriding the BUFSP value in the ACB for the file. The maximum value for nnnnnn is 999999; embedded commas are not permitted.

Usage Notes

1. To display all of the file definitions in effect, enter:

```
dlbl
```

The response will be:

```
ddname DISK fn ft
```

```
  .   .   .   .
  .   .   .   .
  .   .   .   .
```

If no DLBL definitions are in effect, the following message is displayed:

```
DMSDLB324I    No user defined DLBLs in effect
```

2. You may enter an OS or VSE file identification on the DLBL command line. The maximum length of the file identification is 44 characters, including periods. For example, the file TEST.INPUT.SOURCE.D could be identified as follows:

```
dlbl dd1 c dsn test input source d (options...
```

```
-- or --
```

```
dlbl dd1 c dsn test.input.source.d (options...
```

Or, it may be entered interactively, as follows:

```
dlbl dd1 c dsn ? (options)
DMSDLB220R   Enter data set name:
test.input.source.d
```

If the dataset name is entered interactively, the dataset name *must* be entered in its exact form. If it is entered as a command, or from EXEC 2, the dataset name *may* be entered in its exact form. If the command is entered with blanks separating the qualifiers, DLBL replaces them with periods. If it is entered via CMS EXEC, the periods between qualifiers must be omitted, and the qualifiers must be one to eight characters long.

3. In VSE, a VSAM data set that has been defined as DUMMY is opened with an error code of X'11'. CMS supports the DUMMY operand of the DLBL command in the same manner. OS users should not use the DUMMY operand in CMS, since a dummy data set does not return, on open, an end-of-file indication.
4. Do not use the same ddname for a CMS disk or directory if a DLBL already exists with the same ddname for a DOS disk. The use of DSN and CMS is not interchangeable.
5. DLBL uses the extended plist for processing the DSN *qual1 [.qual2...qualn]* parameter. If you are calling DLBL from an assembler language program and using DSN *qual1 [.qual2...qualn]*, you should supply an extended plist. The *VM/SP Application Development Guide for CMS* book has more information on how an assembler language program can supply an extended plist.

Additional Notes for CMS/DOS Users:

1. Each DLBL definition must be associated with a system or programmer logical unit assignment, previously made with an ASSGN command. Specify the SYSxxx option on the first, or only, DLBL definition for a particular ddname. Many DLBL definitions may be associated with the same logical unit. For example:

```
assgn sys100 b
dlbl dd1 b cms test file1 (sys100
dlbl dd2 b cms test file2 (sys100
dlbl dd1 cms test file3
```

is a valid command sequence.

In many situations VSE/VSAM does not require the DLBL command. See the *VSE/VSAM Programmer's Reference*. For information on when the DLBL command is required.

2. The following special ddnames must be used to define VSE private libraries, and must be associated with the indicated logical units:

ddname	Logical Unit	Library
IJSYSSL	SYSSLB	Source statement
IJSYSRL	SYSRLB	Relocatable
IJSYSCL	SYSCLD	Core Image

These libraries must be identified in order to perform librarian functions (with the SSERV, ESERV, DSERV, or RSERV commands) for private libraries; or to link-edit or fetch modules or phases from private relocatable or core image libraries (with the DOSLKED and FETCH commands).

3. Each VSE file has a CMS file identifier associated with it by default; the file name is always FILE and the file type is always the same as the ddname. For example, if you enter a DLBL command for a VSE file MOD.TEST.STREAM as follows:

```
dlbl test c dsn mod.test.stream
```

then you can refer to this data set as FILE TEST when you use the STATE command:

```
state file test
```

When you enter a DLBL command specifying only a ddname and file mode, as follows:

```
dlbl junk a
```

CMS assigns a file identifier of FILE JUNK A1 to the ddname JUNK.

4. The FILEDEF command performs a function similar to that of the DLBL command; you need to use the FILEDEF command in CMS/DOS only:
- When you want to override a default ddname for an assembler input or output file.
 - When you want to use the MOVEFILE command to process a file.
5. If you use the DUMMY operand, you must have issued an ASSGN command specifying a device type of IGN, or ignore, for the SYSxxx unit specified in the DLBL command, for example,

```
assgn sys003 ign
dlbl test dummy (sys003
```

Specifying VSAM Extent Information: You may specify extent information when you use the access method services control statements DEFINE SPACE, DEFINE MASTERCATALOG, DEFINE USERCATALOG, DEFINE CLUSTER (UNIQUE); or when you use the IMPORT or IMPORTRA functions for a unique file.

In many situations, VSE/VSAM does not require EXTENT information. See *VSE/VSAM Programmer's Reference* for information on when EXTENT information is required.

When you enter the EXTENT option of the DLBL command, you are prompted to enter the disk extents for the specified file. You must enter extent information in accordance with the following rules:

- For count-key-data devices, you must specify the starting track number and number of tracks for each extent, as follows:

```
19 38
```

This extent allocates 38 tracks, beginning with the 19th track, on a 3330 device.

- For fixed-block devices, you must specify the starting block number and the number of blocks for each extent. The following example allocates 200 blocks, starting at block number 352, on a fixed-block device.

```
352      200
```

Because VSAM rounds the starting block to the next highest cylinder boundary, it is advisable to specify the starting block on a cylinder boundary.

- All count-key-data extents must begin and end on cylinder boundaries, regardless of whether the AMSERV file contains extent information in terms of cylinders, tracks, or records.
- Multiple extent entries may be entered on a single line separated by commas or on different lines. Commas at the end of a line are ignored.
- Multiple extents for the same volume must be entered in numerically ascending order; for example:

```
20 400, 600 80
```

These extents are valid for a 2314 device.

- When you enter multivolume extents, you must specify the file mode letter and logical unit associated with each disk that contains extents; extents for each disk must be entered consecutively. For example:

```
assgn sys001 b
assgn sys002 c
assgn sys003 d
dlbl file1 b (extent sys001
```

```
DMSDLB331R    Enter extent specifications:
```

```
100 60, 400 80, 60 40 d sys003
200 100 c sys002
400 100 c sys002
      (null line)
```

specifies extents on disks accessed at file modes B, C, and D. These disks are assigned to the logical units SYS001, SYS002, and SYS003. Since B is the file mode specified on the DLBL command line, it does not need to be respecified along with the extent information.

- A DASD volume must be mounted, accessed, and assigned for each disk file mode referenced in an extent.

When you are finished entering extent information, you must enter a null line to terminate the DLBL command sequence. If you do not, an error may result and you will have to reenter the DLBL command. If you make any error entering the extents, you must reenter all the extent information.

The DLBL command does not check the extents to see whether they are on cylinder boundaries or whether they are entered in the proper sequence. If you do not enter them correctly, the access method services DEFINE function will terminate with an error.

CMS assigns sequence numbers to the extents according to the order in which they were entered. These sequence numbers are listed when you use the LISTDS command with the EXTENT option.

In order to display the actual extents that were entered for a VSAM data set at DLBL definition time, the following commands may be entered:

```
DLBL (EXTENT) or QUERY DLBL EXTENT
```

Either of these commands will provide the following information to the user:

DDNAME The VSE file name or OS ddname.

FM	The CMS disk file mode identifying the disk on which the extent resides.
LOGUNIT	The VSE logical unit specification (SYSxxx). This operand will be blank for a data set defined while in CMS/OS environment; that is, the SET DOS ON command had not been issued at DLBL definition time.
EXTENT	Specifies the relative starting track number and number of tracks for each extent entered for the given dataset ddname.

If no DLBL definitions with extent information are active, the following message is issued:

```
DMSDLB324I      No user defined EXTENTS in effect
```

Identifying Multivolume VSAM Extents: When you want to execute a program or use access method services to reference an existing multivolume VSAM data set, you may use the MULT option on the DLBL command that identifies the file.

In many situations, VSE/VSAM does not require this information. See *VSE/VSAM Programmer's Reference* for information on when this type of EXTENT information is required.

When you use the MULT option, you are prompted to enter additional disk file mode letters, as follows:

```
assgn sys001 c
assgn sys002 d
assgn sys003 e
assgn sys004 f
assgn sys005 g
dlbl infile c (mult sys001
```

```
DMSDLB330R      Enter volume specifications:
```

```
d sys002, e sys003 , f sys004
g sys005
      (null line)
```

The above identifies a file that has extents on disks accessed at file modes C, D, E, F, and G. These disks have been assigned to the logical units SYS001, SYS002, SYS003, SYS004, and SYS005. The rules for entering multiple extents are:

- All disks must be mounted, accessed, and assigned when you issue the DLBL command.
- You must not repeat the file mode letter and logical unit of the disk that is entered on the DLBL command line (C in the above example).
- If you enter more than one file mode letter and logical unit on a line, they must be separated by commas; trailing commas on a line are ignored.
- A maximum of nine disks may be specified; you do not need to specify them in alphabetical order.

You must enter a null line to terminate the command when you are finished entering extents; if not, an error may result and you must reenter the entire command sequence.

In order to display the volumes on which all multivolume data sets reside, the following commands are issued:

DLBL (MULT) or QUERY DLBL MULT

The following information concerning multiple volume datasets is provided:

- DDNAME** The VSE file name or OS ddname.
- FM** The CMS disk file mode identifying one of the disks on which the dataset resides.
- LOGUNIT** The VSE logical unit specification (SYSxxx). This operand will be blank for a data set defined while in CMS/OS environment; that is, the SET DOS ON command had not been issued at DLBL definition time.

If no DLBL definitions with multiple volume specifications are active, the following message is issued:

DMSDLB324I No user defined MULTs in effect

Using VSAM Catalogs: There are two special ddnames you must use to identify a VSAM master catalog and job catalog:

- IJSYSCT** identifies the master catalog when you initially define it (using AMSERV), and when you begin a terminal session. You should use the PERM option when you define it.

You must assign the logical unit SYSCAT to the disk on which the master catalog resides. If you are redefining a master catalog that has already been identified, you may omit the SYSCAT option on the DLBL command line.

- IJSYSUC** identifies a job catalog to be used for subsequent AMSERV jobs or VSAM programs.

Any programmer logical unit may be used to assign a job catalog.

Only one VSAM catalog is ever searched when a VSAM function is performed. If a job catalog is defined, you may override it by using the CAT option on the DLBL command for a data set. The following DLBL command sequence illustrates the use of catalogs:

```
assgn syscat c
dlbl ijsysct c dsn mastcat (perm syscat
```

identifies the master catalog, MASTCAT, for the terminal session.

```
assgn sys010 d
dlbl ijsysuc d dsn mycat (perm sys010
```

identifies the job (user) catalog, MYCAT, for the terminal session.

```
assgn sys100 e
dlbl intest1 e dsn test.case (vsam sys100
```

identifies a VSAM file to be used in a program. It is cataloged in the job catalog, MYCAT.

```
assgn sys101 f
dlbl cat3 f dsn testcat (cat ijsysct sys101
```

identifies an additional user catalog, which has an entry in the master catalog. Since a job catalog is in use, you must use the CAT option to indicate that another catalog, in this case the master catalog, should be used.

```
dlbl infile f dsn test.input (cat cat3 sys101
```

identifies an input file cataloged in the user catalog TESTCAT, which was identified with a ddname of CAT3 on the DLBL command.

The selection of a VSAM catalog for AMSERV jobs and VSAM programs running in CMS is summarized in Figure 4.

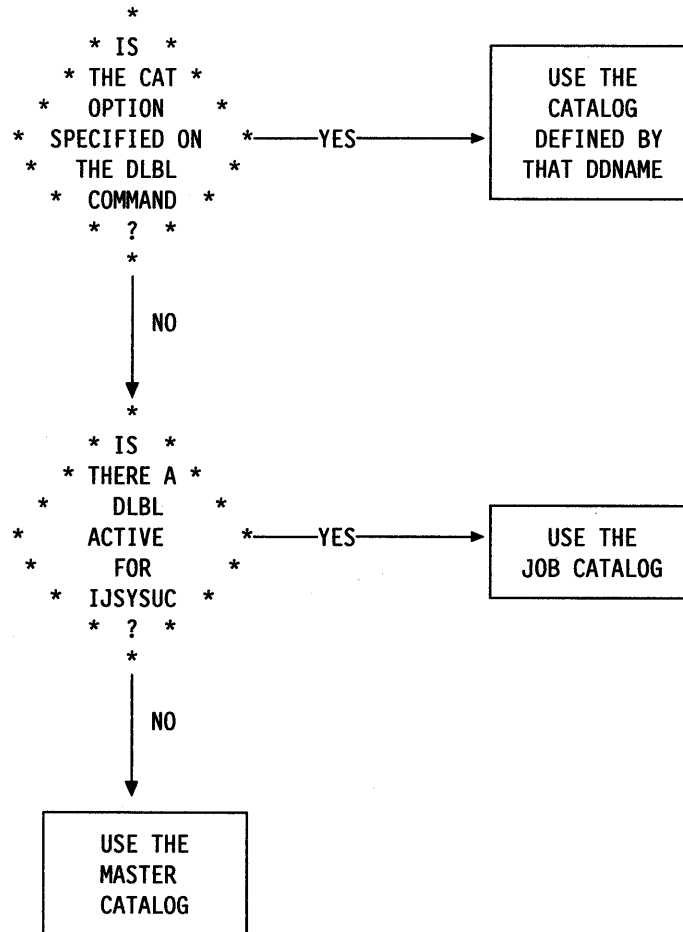


Figure 4. Determining Which VSAM Catalog to Use

Usage Notes for OS VSAM Users:

1. You may use the DLBL command to identify all access method services input and output files, and to identify all VSAM input and output files referenced in programs.

For all other file definitions, including OS or CMS files referenced in programs that use VSAM data management, you must use the FILEDEF command.

File definition statements, either DLBL or FILEDEF, are not always required by VSAM. For more information on file definition requirements, see *VSE/VSAM Programmer's Reference*.

2. A DLBL ddname may have a maximum of seven characters. If you have ddnames in your programs that are eight characters long, only the first seven characters are processed when the programs are executed in CMS. If you have

two ddnames with the same first seven characters and you attempt to execute this program in CMS, you will receive an open error when the second file is opened. You should recompile these programs providing unique seven-character ddnames.

3. If you release a disk or directory for which you have a DLBL definition in effect, you should clear the DLBL definition before you execute a VSAM program or an AMSERV command. CMS checks that all disks or directories for which there are DLBL definitions are accessed, and issues error message DMSSTT069E if any are not.
4. The DLBL command does not support the DISP option. DISP is used in VSE/VSAM to specify the disposition of a reusable file. Therefore, in CMS, only the default values are available. For more information on the DISP option, refer to the *VSE/VSAM Programmer's Reference*.
5. If you access an OS or DOS formatted disk as more than one file mode, the first sequential file mode will be used when VSAM automatically assigns the logical unit, regardless of the file mode specified in the DLBL.

Specifying VSAM Extent Information: You may specify extent information when you use the access method services control statements DEFINE SPACE, DEFINE MASTERCATALOG, DEFINE USERCATALOG, DEFINE CLUSTER (UNIQUE); or when you use the IMPORT or IMPORTRA functions for a unique file. Space allocation is made only for primary allocation amounts.

In many situations, VSE/VSAM does not require EXTENT information. See *VSE/VSAM Programmer's Reference* for information on when EXTENT information is required.

When you enter the EXTENT option of the DLBL command, you are prompted to enter the disk extents for the specified file. You must enter extent information in accordance with the following rules:

- For count-key-data devices, you must specify the starting track number and number of tracks for each extent, as follows:

```
19 38
```

This extent allocates 38 tracks, beginning with the 19th track, on a 3330 device.

- For fixed-block devices, you must specify the starting block number and the number of blocks for each extent. The following example allocates 200 blocks, starting at block number 352, on a fixed-block device.

```
352      200
```

Because VSAM rounds the starting block to the next highest cylinder boundary, it is advisable to specify the starting block on a cylinder boundary.

- All count-key-data extents must begin and end on cylinder boundaries, regardless of whether the AMSERV file contains extent information in terms of cylinders, tracks, or records.
- Multiple extent entries may be entered on a single line separated by commas or on different lines. Commas at the end of a line are ignored.
- Multiple extents for the same volume must be entered in numerically ascending order; for example:

```
20 400, 600 80
```

These extents are valid for a 2314 device.

- When you enter multivolume extents, you must specify the file mode letter for extents on additional disks; extents for each disk must be entered consecutively. For example:

```
dlbl file1 b (extent
DMSDLB331R   Enter extent specifications:
100 60, 400 80, 60 40 d
200 100 c
400 100 c
      (null line)
```

specifies extents on disks accessed at file modes B, C, and D. Since B is the file mode specified on the DLBL command line, it does not need to be re-specified along with the extent information.

- A DASD volume must be mounted and accessed for each file mode referenced in an extent.

When you are finished entering extent information, you must enter a null line to terminate the DLBL command sequence. If you do not, an error may result and you will have to reenter the entire DLBL command. If you make any error entering the extents, you must reenter all the extent information.

The DLBL command does not check the extents to see if they are on cylinder boundaries or that they are entered in the proper sequence. If you do not enter them correctly, the access method services DEFINE function terminates with an error.

CMS assigns sequence numbers to the extents according to the order in which they were entered. These sequence numbers are listed when you use the LISTDS command with the EXTENT option.

Identifying Multivolume VSAM Extents: When you want to execute a program or use access method services to reference an existing multivolume VSAM data set, you may use the MULT option on the DLBL command that identifies the file.

In many situations, VSE/VSAM does not require this information. See *VSE/VSAM Programmer's Reference* for information on when this type of EXTENT information is required.

When you use the MULT option, you are prompted to enter additional disk file mode letters, as follows:

```
dlbl infile c (mult
DMSDLB330R   Enter volume specifications:
d, e, f
g
      (null line)
```

The above example identifies a file that has extents on disks accessed at file modes C, D, E, F, and G. The rules for entering multiple extents are:

- All disks must be mounted and accessed when you issue the DLBL command.
- You must not repeat the file mode letter of the disk that is entered on the DLBL command line (C in the above example).

- If you enter more than one file mode letter on a line, they must be separated by commas; trailing commas on a line are ignored.
- A maximum of nine disks may be specified; you do not need to specify them in alphabetical order.

You must enter a null line to terminate the command when you are finished entering extents; if not, an error may result and you must re-enter the entire command sequence.

Using VSAM Catalogs: There are two special ddnames you must use to identify a VSAM master catalog and job catalog:

- IJSYSCT** identifies the master catalog, both when you initially define it (using AMSERV) and when you begin a terminal session. You should use the PERM option when you define it.
- IJSYSUC** identifies a job catalog to be used for subsequent AMSERV jobs or VSAM programs.

Only one VSAM catalog is ever searched when a VSAM function is performed. If a job catalog is defined, you may override it by using the CAT option on the DLBL command for a data set. The following DLBL command sequence illustrates the use of catalogs:

```
dlbl ijsysct c dsn mastcat (perm
```

identifies the master catalog, MASTCAT, for the terminal session.

```
dlbl ijsysuc d dsn mycat (perm
```

identifies the job (user) catalog, MYCAT, for the terminal session.

```
dlbl intest1 e dsn test.case (vsam
```

identifies a VSAM file to be used in a program. It is cataloged in the job catalog, MYCAT.

```
dlbl cat3 dsn testcat (cat ijsysct
```

identifies an additional user catalog, which has an entry in the master catalog. Since a job catalog is in use, you must use the CAT option to indicate that another catalog, in this case the master catalog, should be used.

```
dlbl infile e dsn test.input (cat cat3
```

identifies an input file cataloged in the user catalog TESTCAT, which was identified with a ddname of CAT3 on the DLBL command.

The selection of a VSAM catalog for AMSERV jobs and VSAM programs running in CMS is summarized in Figure 4.

Responses

If the DLBL command is issued with no operands, the current DLBL definitions are displayed at your terminal:

```
ddname1 device1 [fn1 ft1 fm1 [datasetname1]]
      .       .       .       .       .
      .       .       .       .       .
      .       .       .       .       .
ddnamen devicen [fnn ftn fmn [datasetnamen]]
```

DMSDLB220R Enter data set name:

This message is displayed when you use the DSN ? form of the DLBL command. Enter the exact DOS or OS data set name.

DMSDLB320I Maximum number of disk entries recorded

This message indicates that nine volumes have been specified for a VSAM data set, which is the maximum allowed under CMS.

DMSDLB321I Maximum number of extents recorded

This message indicates that 16 extents on a single disk or minidisk have been specified for a VSAM data space, catalog, or unique data set. This is the maximum number of extents allowed on a minidisk or disk.

DMSDLB322I DDNAME *ddname* not found; no CLEAR executed

This message indicates that the clear function was not performed because no DLBL definition is in effect for the *ddname*.

DMSDLB323I {Job|Master} catalog DLBL cleared

This message indicates that either the master catalog or job catalog has been cleared as a result of a clear request.

You also receive this message if you issue a DLBL * CLEAR command, and any DLBL definition is in effect for IJSYSCT or IJSYSUC that was not entered with the PERM option.

DMSDLB330R Enter volume specifications:

This message prompts you to enter volume specifications for existing multivolume VSAM files. (See "Identifying Multivolume VSAM Extents" in the appropriate usage section.)

DMSDLB331R Enter extent specifications:

This message prompts you to enter the data set extent or extents of a new VSAM data space, catalog or unique data set. (See "Specifying VSAM Extent Information" in the appropriate usage section.)

Messages and Return Codes

DMSDLB001E	No filename specified [RC = 24]
DMSDLB003E	Invalid option: <i>option</i> [RC = 24]
DMSDLB005E	No <i>option</i> specified [RC = 24]
DMSDLB023E	No filetype specified [RC = 24]
DMSDLB048E	Invalid mode <i>mode</i> [RC = 24]
DMSDLB050E	Parameter missing after DDNAME [RC = 24]
DMSDLB065E	<i>option</i> option specified twice [RC = 24]
DMSDLB066E	<i>option1</i> and <i>option2</i> are conflicting options [RC = 24]

) | DMSDLB069E Filemode *mode* not accessed [RC = 36]
| DMSDLB070E Invalid parameter *parameter* [RC = 24]
| DMSDLB086E Invalid DDNAME *ddname* [RC = 24]
| DMSDLB109S Virtual storage capacity exceeded [RC = 104]
| DMSDLB221E Invalid data set name [RC = 24]
| DMSDLB301E *SYSaaa* not assigned for filemode *fm* [RC = 36]
| DMSDLB302E No *SYSXXX* operand entered [RC = 24]
| DMSDLB304E Invalid operand value *value* [RC = 24]
| DMSDLB305E Incomplete extent range [RC = 24]
| DMSDLB306E *SYSaaa* not assigned for IGNORE [RC = 36]
| DMSDLB307E Catalog DDNAME *ddname* not found [RC = 24]
| DMSDLB308E *mode* filemode in [non-]CMS format; invalid for [non-]CMS
| dataset [RC = 24]

DOSLIB

Use the DOSLIB command to delete, compact, or list information about the executable phases in a CMS/DOS phase library.

Format

DOSLIB	$\left\{ \begin{array}{l} \text{DEL } \textit{libname} \textit{ phasename1} \textit{ [...phasenamen]} \\ \text{COMP } \textit{libname} \\ \text{MAP } \textit{libname} \textit{ [(options...)]} \end{array} \right\}$
	<u>Options:</u> $\left[\begin{array}{l} \text{TERM} \\ \text{DISK} \\ \text{PRINT} \end{array} \right]$

Operands**DEL**

deletes phases from a CMS/DOS phase library. The library is not erased when the last phase is deleted from the library.

COMP

compacts a CMS/DOS phase library.

MAP

lists certain information about the phases of a DOSLIB. Available information provided is phase name, size, and relative location in the library.

libname

is the file name of a CMS/DOS phase library. The file type must be DOSLIB.

phasename1...phasenamen

is the name of one or more phases that exist in the CMS/DOS phase library.

MAP Options

The following options specify the output device for the MAP function. If more than one option is specified, only the first option is used.

TERM

displays the MAP output at the terminal.

DISK

writes the MAP output to a file on the disk or directory accessed as A with the file identifier of 'libname MAP A5'. If a file with that name already exists, the old file is erased.

PRINT

spools the MAP output to the virtual printer.

Usage Notes

1. The CMS/DOS environment does not have to be active when you issue the DOSLIB command.
2. Phases may only be added to a DOSLIB by the CMS/DOS linkage editor as a result of the DOSLKED command.
3. To fetch a program phase from a DOSLIB for execution, you must issue the GLOBAL command to identify the DOSLIB. When a FETCH command or dynamic fetch from a program is issued, all current DOSLIBs are searched for the specified phases.
4. If DOSLIBs are very large, or there are many of them to search, program execution is slowed down accordingly. To avoid excessive execution time, you should keep your DOSLIBs small and issue a GLOBAL command specifying only those libraries that you need.

Example

To compact MYLIB DOSLIB, you would enter the command:

```
doslib comp mylib
```

Responses

When you use the TERM option on the DOSLIB MAP command line, the following is displayed:

PHASE	INDEX	BLOCKS
name1	loc	size
.	.	.
.	.	.
.	.	.

Messages and Return Codes

DMSDSL002E	File <i>fn</i> DOSLIB not found [RC = 28]
DMSDSL003E	Invalid option: <i>option</i> [RC = 24]
DMSDSL013W	Phase <i>phase</i> not found in library <i>libname</i> [RC = 4]
DMSDSL014E	Invalid function <i>function</i> [RC = 24]
DMSDSL037E	Filemode <i>mode</i> [(<i>vdev</i>)] is accessed as read/only [RC = 36]
DMSDSL046E	No library name specified [RC = 24]
DMSDSL047E	No function specified [RC = 24]
DMSDSL069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSDSL070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSDSL098E	No phase name specified [RC = 24]
DMSDSL104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]
DMSDSL105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSDSL213W	Library <i>fn ft fm</i> not created [RC = 4]
DMSDSL213W	Library <i>libname</i> has no members [RC = 4]
DMSDSL653E	Error executing <i>command</i> rc = <i>nn</i> [RC = 40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in copying a file	70
Errors in erasing a file	145

DOSLKED

Use the DOSLKED command in CMS/DOS to link-edit TEXT files from CMS disks, SFS directories, or object modules from VSE private or system relocatable libraries and place them in executable form in a CMS phase library (DOSLIB).

Format

DOSLKED	$fn \left[\begin{array}{c} libname \\ \underline{fn} \end{array} \right] [(options... \]]$
	<u>Options:</u> $\left[\begin{array}{c} \underline{DISK} \\ \underline{PRINT} \\ \underline{TERM} \end{array} \right]$

Operands*fn*

specifies the name of the source file or module to be link-edited. CMS searches for:

1. A CMS file with a file type of DOSLNK
2. A module in a private relocatable library (if IJSYSRL has been defined)
3. A CMS file with a file type of TEXT
4. A module in the system relocatable library (if a mode was specified on the SET DOS ON command line)

libname

designates the name of the DOSLIB where the link-edited phase is to be written. The file type is DOSLIB. If libname is not specified, the default is fn. The output file mode of the DOSLIB is determined as follows:

- If libname DOSLIB exists on a read/write disk or directory, that file mode is used and the output is appended to it.
- If fn DOSLNK exists on a read/write disk or directory, libname DOSLIB is written to that disk or directory.
- If fn DOSLNK exists on a read-only extension of a read/write disk or directory libname DOSLIB is written to the parent.
- If none of the above apply, libname DOSLIB is written to your disk or directory accessed as A.

Options

Only one of the following options should be specified. If more than one is specified, only the first entry is used.

DISK

writes the linkage editor map produced by the DOSLKED command on your disk or directory accessed as A into a file with the file name of *fn* and a file type of MAP. This is the default option.

PRINT

spools the linkage editor map to the virtual printer.

TERM

displays the linkage editor map at your terminal.

Note: All error messages are sent to the terminal as well as to the specified device.

Usage Notes

1. You can create a CMS file with a file type of DOSLNK to contain linkage editor control statements and, optionally, CMS text files.
2. If you want to link-edit a module from a private relocatable library, you must issue an ASSGN command for the logical unit SYSRLB and enter a DLBL command using a ddname of IJSYSRL to identify the library:


```
assgn sysrlb c
dlbl ijsysrl c dsn reloc lib (sysrlb
```

If you have defined a private relocatable library but do not want it to be searched, enter:

```
assgn sysrlb ign
```

to temporarily bypass it.
3. CMS TEXT files may also contain linkage editor control statements INCLUDE, PHASE, and ENTRY. The ACTION statement is ignored when a TEXT file is link-edited.
4. To access modules on a VSE system residence volume, you must have specified the mode letter of the system residence on the SET DOS ON command line:


```
set dos on z
```
5. The search order that CMS uses to locate object modules to be link-edited is:
 - a. The specified object module on the VSE private relocatable library, if one is available
 - b. CMS disks and SFS directories for a file with the specified file name and with a file type of TEXT
 - c. The specified object module on the VSE system relocatable library, if it is available
6. When a phase is added to an existing DOSLIB, it is always written at the end of the library. If a phase that is being added has the same name as an existing phase, the DOSLIB directory is updated to point to the new phase. The old phase is not deleted, however; you should issue the DOSLIB command with the COMP option to compress the space.

If you run out of space in a DOSLIB while you are executing the DOSLKED command, you should reissue the DOSLKED command specifying a different DOSLIB, or compress the DOSLIB before attempting to reissue the DOSLKED command.

7. Prior to performing a DOSLKED on a TEXT file having multiple phase cards following the TEXT END cards, rename the file type of TEXT to a file type DOSLNK.
8. If the input to the DOS linkage editor contains a text file produced by punching a member from a TXTLIB, the last two records are LDT records. When using OS linkage edit, these are recognized as end cards. However, when you are using the DOSLKED command, delete these two cards prior to issuing the command. Failure to do so may cause unpredictable results.

Linkage Editor Control Statements: The CMS/DOS linkage editor recognizes and supports the VSE linkage editor control statements ACTION, PHASE, ENTRY, and INCLUDE. The CMS/DOS linkage editor ignores:

- The SVA operand of the PHASE statement
- The F+ address form for specifying origin on the PHASE statement
- The BG, Fn, and SMAP operands of the ACTION statement

The S-form of specifying the origin on the PHASE statement corresponds to the CMS user area under CMS/DOS. If a default PHASE statement is required, the origin is assumed to be S. The PBDY operand of the PHASE statement indicates that the phase is link-edited on a 4K page boundary under CMS/DOS as opposed to a 2K page boundary for VSE.

In VSE, an ACTION CLEAR control statement clears the unused portion of the core image library to binary zeros. In VSE, the core image library has a defined size, while in CMS/DOS the CMS phase library varies in size, depending on the number of phases cataloged. Therefore, in CMS/DOS an ACTION CLEAR control statement clears the current buffers to binary zeros before loading them; CMS/DOS cannot clear the entire unused portion of the CMS phase library because that portion varies as phases are added to and deleted from the CMS phase library. In CMS/DOS if you want your phases cleared you must issue an ACTION CLEAR control statement each time you add a phase to the CMS phase library.

Linkage Editor Card Types: The input to the linkage editor can consist of six card types, produced by a language translator or a programmer. These cards 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 of text made by the programmer
END	End of module

CMS/DOS supports these six card types in the same manner that VSE does.

Example

To link-edit the TEST TEXT file and place it in the TESTLIB DOSLIB, enter the command:

```
doslked test testlib
```

Responses

When you use the TERM option of the DOSLKED command, the linkage editor map is displayed at the terminal.

```
2101I INVALID OPERATION IN CONTROL STATEMENT
```

This message indicates that a blank card was encountered in the process of link-editing a relocatable module. This message also appears in the MAP file. The invalid card is ignored and processing continues.

Messages and Return Codes

DMSDLK001E	No filename specified [RC = 24]
DMSDLK003E	Invalid option: <i>option</i> [RC = 24]
DMSDLK006E	No read/write filemode accessed [RC = 36]
DMSDLK007E	File <i>fn ft fm</i> is not fixed, 80-character records [RC = 32]
DMSDLK070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSDLK099E	CMS/DOS environment not active [RC = 40]
DMSDLK104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]
DMSDLK105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSDLK210E	Library <i>fn ft</i> is on a read/only filemode [RC = 36]
DMSDLK245S	Error <i>nnn</i> on printer [RC = 100]

DROPBUF

DROPBUF

Use the DROPBUF command to eliminate only the most recently created program stack buffer or a specified program stack buffer and all buffers created after it.

Format

DROPBUF	<i>n</i>
----------------	----------

Operands

n
indicates the number of the first program stack buffer you want to drop. CMS drops the indicated buffer and all buffers created after it. If *n* is not specified, only the most recently created buffer is dropped.

Usage Note

Note that you can specify a number with DROPBUF. For example, if you issue:
DROPBUF 4

CMS eliminates program stack buffer 4 and all program stack buffers created after it. Thus, if there were presently six program stack buffers, CMS would eliminate program stack buffers 6, 5, and 4. If you issued DROPBUF without specifying *n*, only program stack buffer 6 would be eliminated.

Return Codes

If an error occurs in DROPBUF processing, the return code will be one of the following:

Code	Meaning
1	Invalid number specified
2	Specified buffer does not exist

DSERV

Use the DSERV command in CMS/DOS to obtain information that is contained in VSE private or system libraries.

Format

DSERV	$\left\{ \begin{array}{l} \text{CD} \\ \text{RD} \\ \text{SD} \\ \text{PD} \\ \text{TD} \\ \text{ALL} \end{array} \right. \left[\text{PHASE} \left\{ \text{name} \left[\begin{array}{l} \text{nn} \\ \underline{12} \end{array} \right] \right\} \right] \left[\text{d2...dn} \right] \left[(\text{options...}) \right]$
	<p><u>Options:</u> $\left[\begin{array}{l} \text{DISK} \\ \text{TERM} \\ \text{PRINT} \end{array} \right]$ $[\text{SORT}]$</p>

Operands

CD
RD
SD
PD
TD
ALL

specifies that information concerning one or more types of directories is to be displayed or printed. The directory types that can be specified are: CD (core image library), RD (relocatable library), SD (source statement library), PD (procedure library), TD (transient directory), and ALL (all directories).

There is no default value. The private libraries take precedence over system libraries.

PHASE *name*

specifies the name of the phase to be listed. If the phasename ends with an asterisk, all phases that start with the letters preceding the asterisk are listed. This operand is valid only for CD.

nn

is the displacement within the phase where the version and level are to be found (the default is 12).

$[\text{d2...dn}]$

indicates additional libraries whose directories are to be listed. (See Usage Note 1.)

DSERV

Options

DISK

writes the output on the disk or directory accessed as A to a file named DSERV MAP A5. This is the default value if TERM or PRINT is not specified.

TERM

displays the output at your terminal.

PRINT

spools the output to the system printer.

SORT

sorts the entries for each library alphanumerically. Otherwise, the order is the order in which the entries were cataloged, except in the case of the Core Image Library, which is always written alphanumerically by VSE for performance reasons.

Usage Notes

1. You may specify more than one directory on DSERV command line; for example:

```
dserv rd sd cd phase $$bopen (term
```

displays the directories of the relocatable and source statement libraries, as well as the entry for the phase \$\$BOPEN from the core image directory.

You can specify only one phasename or phasename* at a time. However, if you specify more than one PHASE operand, only the last one entered is listed. For example, if you enter:

```
dserv cd phase cor* phase idc*
```

the file DSERV MAP contains a list of all phases that begin with the characters IDC. The first phasename specification is ignored.

2. If you want to obtain information from the directories of private source statement library directories, relocatable library directories, or core image library directories, the libraries must be assigned and identified (via ASSGN and DLBL commands) when the DSERV command is issued. Otherwise, the system library directories are used. System directories are made available when you specify a mode letter on the SET DOS ON command line.
3. The current assignments for logical units are ignored by the DSERV command; output is directed only to the output device indicated by the option list.

Example

To display at your terminal an alphanumeric list of procedures cataloged on the system procedure library, you would issue:

```
dserv pd (sort term
```

Responses

When you use the TERM option of the DSERV command, the contents of the specified directory are displayed at your terminal.

Messages and Return Codes

DMSDSV003E Invalid option: *option* [RC = 24]
DMSDSV021W No transient directory [RC = 4]
DMSDSV022W No core image directory [RC = 4]
DMSDSV023W No relocatable directory [RC = 4]
DMSDSV024W No procedure directory [RC = 4]
DMSDSV025W No source statement directory [RC = 4]
DMSDSV026W *phase* not in library [RC = 4]
DMSDSV027E Invalid device *devtype* [for SYSaaa] [RC = 24]
DMSDSV027W No private core image library [RC = 4]
DMSDSV028W No {private|system} transient directory entries [RC = 4]
DMSDSV047E No function specified [RC = 24]
DMSDSV065E *option* option specified twice [RC = 24]
DMSDSV066E *option1* and *option2* are conflicting options [RC = 24]
DMSDSV070E Invalid parameter *parameter* [RC = 24]
DMSDSV095E Invalid address *vstor* [RC = 100]
DMSDSV099E CMS/DOS environment not active [RC = 40]
DMSDSV105S Error *nn* writing file *fn ft fm* on disk or directory [RC = 100]
DMSDSV245S Error *nnn* on printer [RC = 100]
DMSDSV411S Input error code *nn* on SYSaaa [RC = *rc*]

EDIT

Use the EDIT command to invoke the VM/SP System Product Editor in CMS editor (EDIT) migration mode. Use the editor to create, modify, and manipulate CMS files. In EDIT migration mode, you may execute both EDIT and XEDIT subcommands. For complete details on EDIT migration mode, refer to the *VM/SP System Product Editor Command and Macro Reference*.

To invoke only the CMS editor, refer to the “Usage Note” below.

Once the CMS editor has been invoked, you may only execute EDIT subcommands and EDIT macro requests, and enter data lines into the file. A limited number of CMS commands may be executed in the CMS subset mode. Enter CMS subset mode from the edit environment by issuing the EDIT subcommand, CMS.

You can return control to the CMS environment by issuing the EDIT subcommands FILE or QUIT.

Format

Edit	$fn\ ft\ [fm]\ \ [options...[]]$ <div style="text-align: center; margin-left: 100px;">*</div>
	<u>Options:</u> [LRECL nn] [NODISP]

Operands

fn ft
 is the file name and file type of the file to be created or edited. If a file with the specified file name and file type does not exist, the CMS editor assumes that you want to create a new file, and after you issue the INPUT subcommand, all data lines you enter become input to the file. If a file with the specified file name and file type exists, you may issue EDIT subcommands to modify the specified file.

fm
 is the file mode of the file to be edited, indicating the disk or directory on which the file resides. The editor determines the file mode of the edited file as follows:

Editing existing files: If the file does not reside on your disk or directory accessed as A or its extensions, you must specify fm.

When you specify fm, the specified disk or directory and its extensions are searched. If a file is found on a read-only extension, the file mode of the parent is saved; when you issue a FILE or SAVE subcommand, the modified file is written to the parent.

If you specify fm as an asterisk (*) all accessed disks and directories are searched for the specified file.

Creating new files: If you do not specify fm, the new file is written on your disk or directory accessed as A when you issue the FILE or SAVE subcommands.

Options

LRECL *nn*

is the record length of the file to be created or edited. Use this option to override the default values supplied by the editor, which are determined as follows:

Editing Existing Files: Existing record length is kept regardless of format. If the file has variable-length records and the existing record length is less than the default record length, the default record length is used.

Creating New Files: All new files have a record length of 80, with the following exceptions:

File type	LRECL
LISTING	121
SCRIPT,VSBDATA	132
FREEFORT	81

The maximum record length supported by the editor is 160 characters.

NODISP

forces a 3270 display terminal into line (typewriter) mode. When the NODISP option is in effect, all subcommands that control the display as a 3270 terminal such as SCROLL, SCROLLUP, and FORMAT (and CHANGE with no operands) are made invalid for the edit session.

Note: It is recommended that the NODISP option always be used when editing on a 3066.

Usage Note

When you issue the EDIT command, an EXEC named EDIT EXEC S2 is executed. This EXEC invokes the VM/SP System Product Editor in EDIT migration mode.

If you want to invoke only the CMS editor on a permanent basis, your system programmer must rename this EXEC. Then, when you issue the EDIT command, the EXEC will not execute and the CMS editor will be invoked.

If you want to invoke the CMS editor only for a particular edit session, specify OLD on the EDIT command line. CMS passes the OLD parameter to EDIT EXEC S2 and only the CMS editor is invoked. Note that the old editor has not been enhanced for VM/SP and will not be enhanced for future releases; specifically the old editor will not include any support for new display devices.

Example

If you want to create a new file on your disk or directory accessed as A called OVERTIME DATA, you would enter:

```
edit overtime data a
```

Responses

NEW FILE:

The specified file does not exist.

EDIT:

The edit environment is entered. You may issue any valid EDIT subcommand or macro request.

INPUT:

EDIT

The input environment is entered by issuing the EDIT subcommands REPLACE or INPUT with no operands. All subsequent input lines are accepted as input to the file.

Messages and Return Codes

DMSEDI003E	INVALID OPTION <i>option</i> [RC = 24]
DMSEDI024E	FILE <i>fn ft fm</i> ALREADY EXISTS [RC = 28]
DMSEDI029E	INVALID PARAMETER <i>parameter</i> IN THE 'LRECL' OPTION FIELD [RC = 24]
DMSEDI044E	RECORD LENGTH EXCEEDS ALLOWABLE MAXIMUM [RC = 32]
DMSEDI048E	INVALID MODE <i>mode</i> [RC = 24]
DMSEDI054E	INCOMPLETE FILEID SPECIFIED [RC = 24]
DMSEDI069E	DISK <i>mode</i> NOT ACCESSED [RC = 36]
DMSEDI076E	ACTUAL RECORD LENGTH EXCEEDS THAT SPECIFIED [RC = 40]
DMSEDI104S	ERROR <i>nn</i> READING FILE <i>fn ft fm</i> FROM DISK [RC = 100]
DMSEDI105S	ERROR <i>nn</i> WRITING FILE <i>fn ft fm</i> ON DISK [RC = 100]
DMSEDI117S	ERROR WRITING TO DISPLAY TERMINAL [RC = 100]
DMSEDI132S	FILE <i>fn ft fm</i> TOO LARGE [RC = 88]
DMSEDI143S	UNABLE TO LOAD MODULE [RC = 40]
DMSEDI144S	REQUESTED FILE IS IN ACTIVE STATUS
DMSEDI151E	3278 MOD5 DISPLAY TERMINAL NOT SUPPORTED BY OLD CMS EDITOR
DMSEDX069E	DISK <i>mode</i> NOT ACCESSED [RC = 36]
DMSEDX109S	VIRTUAL STORAGE CAPACITY EXCEEDED [RC = 104]

ERASE

Use the ERASE command to delete:

- one or more CMS files from a read/write disk
- one or more CMS files from a Shared File System (SFS) directory accessed as read/write
- one or more CMS files in a Shared File System (SFS) directory for which you have write authority
- one of your SFS directories.

Format

ERASE	$\left\{ \begin{array}{l} \left\{ \begin{array}{l} \{fn\} \\ \{*\} \end{array} \right\} \left\{ \begin{array}{l} \{ft\} \\ \{*\} \end{array} \right\} \left[\begin{array}{l} fm \\ dirid \\ * \end{array} \right] \left[(\text{optionA... } []) \right] \\ \\ dirid \qquad \qquad \qquad [(\text{optionA optionB...} [])] \end{array} \right\}$
	<p>OptionA: $\left[\begin{array}{l} \text{Type} \\ \text{Notype} \\ \text{STACK} \left[\begin{array}{l} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{FIFO} \\ \text{LIFO} \end{array} \right]$</p> <p>OptionB: $\left[\begin{array}{l} \text{FILEs} \\ \text{NOFILEs} \end{array} \right]$</p>

Operands

fn

is the name of the file(s) to be erased. An asterisk (*) coded in this position indicates that all names are to be used.

ft

is the file type of the file(s) to be erased. An asterisk (*) coded in this position indicates that all file types are to be used.

fm

is the file mode of the files to be erased. If this field is omitted, only the disk or directory accessed as A is searched. An asterisk (*) coded in this position indicates that files with the specified name and/or file type are to be erased from all disks and directories that are accessed as read/write. If an asterisk is entered as the file mode, then either the file name or the file type or both must be specified by name.

ERASE

dirid

is either:

- the name of the directory containing the files to be erased, if you specify *fn ft*. If this field is omitted, only the disk or directory accessed as A is searched.
- or
- the name of the directory to be erased, if you do not specify *fn ft*. You must own a directory to erase it.

See "Naming Shared File System (SFS) Directories" on page 4 for a description of *dirid*.

OptionA

Type

displays at the terminal the file identifier of each file erased or the directory identifier if you are erasing an SFS directory.

Notype

file or directory identifiers are not displayed at the terminal. NOTYPE is the default.

STACK [FIFO]

STACK LIFO

places the information in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

OptionB

FILEs

indicates that you want all the files in the directory to be erased along with the directory. If the directory contains any subdirectories, the directory is not erased. This option is only valid for erasing directories in your own directory structure.

NOFiles

indicates that the directory must be empty of all files and subdirectories before it can be erased. It can contain erased and revoked aliases. NOFILES is the default.

Usage Notes

1. If you specify an asterisk for both file name and file type, you must specify both a file mode letter and a number. For example:
erase * * a5
2. To erase all files on a particular minidisk, you can access the minidisk using the ACCESS command with the ERASE option. Another way to erase all files from a minidisk is to use the FORMAT command to reformat it.
3. You can erase another user's base file if you have write authority to both the file and the directory containing the file. You can erase another user's alias if you

have read authority to the base file and write authority to the directory containing the alias. When erasing base files or aliases in another user's directory, the *fm* form of *dirid* is not valid. See "Naming Shared File System (SFS) Directories" on page 4 for information on the forms of *dirid*.

4. When you erase a base file in an SFS directory:
 - the file is erased
 - aliases of the base file are turned into erased aliases
 - all authorities to use the file are revoked.
5. When you erase an alias of a base file:
 - only the specified alias is erased
 - base file and other aliases of the base file are not affected
 - lock status of the base file has no effect.
6. A directory can only be erased by the directory owner or an administrator.
7. You cannot erase a file in an SFS directory or erase an SFS directory if:
 - the file is open by the Open program function, DMSOPEN, or if you have opened the directory with the Open Directory program function, DMSOPDIR.
 - the file or directory is locked by another user
 - you or any user has a SHARE lock on the file or directory.
8. You can erase a file or directory for which you have an UPDATE or EXCLUSIVE lock.
9. If another user has your directory open using the DMSOPDIR program function, either you or your administrator can erase the open directory. Also, an administrator can erase a directory that the owner has open.
10. You may also erase your open directory if the following conditions are met:
 - you opened the directory with an intent of FILE
 - you have not specified the FILES option on the ERASE command
11. You can erase files from an open directory as long as you do not use special characters to specify a set of files.
12. To erase a directory that has files in it other than erased or revoked aliases, you must specify the FILES option.
13. After a directory is erased, any user that had the directory accessed will receive an error when attempting to work with a file in that directory.
14. If you specify TYPE or one of the STACK options when you erase a directory, only the directory identifier is listed. No file identifiers are displayed or stacked.
15. If asterisks are used to erase a set of files and an error occurs, the last non-zero return code is returned. If a fatal error (such as a permanent I/O error) occurs on a disk, an error message is issued and the virtual machine enters a disabled wait state.
16. You can invoke the ERASE command from the terminal, from an exec file, or as a function from a program. No error messages are issued if ERASE is invoked:
 - As a function from a program

- From a CMS EXEC file that has the &CONTROL NOMSG option in effect (only stops messages 002, 1210, and 1184 from appearing)
 - From an EXEC2 exec file and CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect.
17. You can issue ERASE from the command (Cmd) column on any of the FILELIST screens if you are authorized to erase the file or directory. For example, to erase a file that is located in another user's directory that is displayed on the screen, enter:

```
erase /ntd
```

where

nt

represents name, and type - the *fn ft* that is displayed.

d

represents directory - the directory name that is displayed.

To erase a file in your own directory, enter:

```
erase /
```

18. Program calltype restrictions

Program calls to the ERASE command have calltype restrictions. Extended ERASE command syntax is supported only when the CMSCALL calltype is X'0B' (command) or X'01' (EXEC 2 or System Product Interpreter ADDRESS COMMAND call). See the CMSCALL macro in the *Application Development Reference for CMS*.

Specifically,

- Valid file modes, including an asterisk or a blank, are supported for all calltypes.
- Input to the Shared File System (i.e., directory IDs other than a valid file mode, an asterisk, blank, or the ERASE *dirid* form of the command) are only supported for calltypes of X'01' and X'0B'. If anything other than a valid file mode is specified with an unsupported calltype, a parsing error (RC=24) will occur.
- Options are respected only for calltypes X'01' and X'0B'-X'0E'. Also the TYPE option is not respected for X'01'. If options are specified with calltypes other than X'01' and X'0B' - X'0E', they are ignored.
- No extended syntax or options are supported for branch entries to DMSERS.

Note: If the calltype is not X'01' or X'0B', you can still erase files from directories that are accessed as read/write by using the directory's file mode in the call to the ERASE command.

Use the X'0B' calltype when messages and the TYPE option are desired. Use the X'01' calltype when you want to suppress messages and any output from the TYPE option.

Example

To erase all the files on your minidisk or directory accessed as A with a file type of DATA, you would enter:

```
erase * data a
```

Responses

If you specify the TYPE option, the file identifier of each file erased is displayed. For example:

```
erase oldfile temp (type
```

results in the display:

```
OLDFILE TEMP A1
Ready;
```

Messages and Return Codes

DMSERA002E	File [<i>fn</i> [<i>ft</i> [<i>fm</i> <i>dirname</i>]]] not found [RC=28]
DMSERA109E	Virtual storage capacity exceeded [RC=25]
DMSERA1160E	Directory <i>dirname</i> already open. [RC=70]
DMSERA1161E	Directory <i>dirname</i> contains subdirectories and thus cannot be erased. [RC=40]
DMSERA1162E	Directory <i>dirname</i> is not empty; specify FILES option [RC=40]
DMSERA1163E	The ERASE command failed for <i>fn ft fm dirname</i> [RC= <i>nn</i>]
DMSERA1180E	You have an explicit lock on file <i>fn ft fm dirname</i> ; the erase failed [RC=70]
DMSERA1180E	You have an explicit lock on directory <i>dirname</i> or on an object in the directory; the erase failed [RC=70]
DMSERA1181E	Directory <i>dirname</i> contains an open file and thus cannot be erased [RC=70]
DMSERA1184E	File <i>fn ft fm dirname</i> not found or you are not authorized for it [RC=28]
DMSERA1184E	Directory <i>dirname</i> not found or you are not authorized for it [RC=28]
DMSERA1197E	Directory <i>dirname</i> could not be opened; no files erased [RC=70]
DMSERA1198E	File <i>fn ft fm dirname</i> is currently open; it must be closed before it can be erased [RC=70]
DMSERA1198E	Directory <i>dirname</i> is currently open; it must be closed before it can be erased [RC=70]
DMSERA1199E	You cannot erase a top directory [RC=88]
DMSERA1210E	Directory <i>dirname</i> not found [RC=28]
DMSERS037E	Filemode <i>modeis</i> accessed as read/only [RC=36]
DMSERS048E	Invalid mode <i>mode</i> [RC=24]
DMSERS054E	Incomplete fileid specified [RC=24]
DMSERS069E	Filemode <i>mode</i> not accessed [RC=36]
DMSERS070E	Invalid parameter <i>parameter</i> [RC=24]
DMSERS071E	Erase * * [<i>fm</i> *] not allowed [RC=24]
DMSJED1187E	Too many subdirectory levels in <i>dirid</i> [RC=24]
DMSJED1188E	Filemode <i>fm</i> is not associated with a directory [RC=74]
DMSJED1189E	Filemode <i>fm</i> is associated with a top directory [RC=24]
DMSJED1223E	There is no default file pool currently defined [RC=40]

Additional system messages may be issued by this command. The ERASE calltype affects the message you may get as noted below. The reasons for these messages, the calltype, and the page that lists them are:

ERASE

Reason	Calltype	Page
Errors in command syntax	X'0B' - X'0E'	811
Errors in finding a file	X'0B' - X'0D'	601
Errors in the Shared File System	X'0B' - X'0E'	813

ESERV

Use the **ESERV EXEC** procedure in CMS/DOS to copy edited VSE macros from system or private source statement E sublibraries to CMS files, or to list de-edited macros.

Format

ESERV	<i>fn</i>
--------------	-----------

Operands

fn specifies the file name of the CMS file that contains the **ESERV** control statements; it must have a file type of **ESERV**. The logical unit **SYSIPT** must be assigned to the disk or directory on which the **ESERV** file resides. The file name is the *fn* of the **LISTING** and **MACRO** files produced by the **ESERV** program.

Usage Notes

1. The input file can contain any or all of the **ESERV** control statements as defined in *Guide to the DOS/VSE Assembler*.
2. You must have a read/write disk or directory accessed as **A** when you use the **ESERV** command.
3. To copy macros from the system source statement library, you must have entered the CMS/DOS environment specifying the mode letter of the VSE system residence. To copy from a private source statement library, you must assign the logical unit **SYSSLB** and issue a **DLBL** command for the **ddname IJSYSSL**.
4. The output of the **ESERV** program is directed (as in **VSE/AF**) to devices assigned to the logical units **SYSLST** and/or **SYSPCH**. If either **SYSLST** or **SYSPCH** is not assigned, the following files are created:

<i>Unit</i>	<i>Output File</i>
SYSLST	<i>fn</i> LISTING mode
SYSPCH	<i>fn</i> MACRO mode

where mode is the mode letter of the disk or directory on which the source file, *fn* **ESERV** resides. If *fn* **ESERV** is on a read-only disk or directory, the files are written to your disk or directory accessed as **A**.

You can override default assignments made by the **ESERV EXEC** as follows:

- If you assign **SYSIPT** to **TAPE** or **READER**, the source statements are read from that device.
 - If you assign **SYSLST** or **SYSPCH** to another device, the **SYSLST** or **SYSPCH** files are written to that device.
5. The **ESERV EXEC** procedure clears all **DLBL** definitions, except those entered with the **PERM** option.
 6. If you want to use the **ESERV** command in an **EXEC** procedure, you must use the **EXEC** command (because **ESERV** is also an **EXEC**).

7. When you use the ESERV control statements PUNCH or DSPCH, the ESERV program may generate CATAL.S, END, or /* records in the output file. When you add a MACRO file containing these statements to a CMS macro library using the MACLIB command, the statements are ignored and are not read into the MACLIB member.
8. Any disks or directories accessed with a mode letter of "R" or "T" should be in read/only mode when an ESERV is running on them, otherwise message DMSDLB301E may occur.
9. If you want to issue ESERV from an EXEC program, you should precede it with the EXEC command; that is, specify

```
exec eserv
```
10. If a MACRO or LISTING file with the specified file name exists before the ESERV command is issued, the file(s) are renamed:
 - *fn* CMSUT1 for the MACRO file
 - *fn* CMSUT2 for the LISTING file

This preserves the original file authorities. If an error occurs these temporary files may be left on your disk or directory. You may want to rename them to your original MACRO or LISTING file.

File authorities are not maintained if you specify a DLBL with a file type other than MACRO or LISTING for the output macro or listing files.

Responses

None. The CMS ready message indicates that the ESERV program completed execution successfully. You may examine the SYSLST output to verify the results of the ESERV program execution.

Messages and Return Codes

DMSERV001E NO FILENAME SPECIFIED. [RC = 24]
 DMSERV002E FILE *fn* *ESERV* NOT FOUND. [RC = 28]
 DMSERV006E No read / write filemode accessed. [RC = 36]
 DMSERV070E INVALID ARGUMENT *argument* [RC = 24]
 DMSERV099E CMS/DOS ENVIRONMENT NOT ACTIVE. [RC = 40]
 DMSERV027E INVALID DEVICE *devtype* FOR *SYSaaa* [RC = 24]
 DMSERV037E Filemode *mode* is read only. [RC = 36]

Note: The ESERV EXEC calls other CMS commands to perform certain functions, and so you may receive error messages that occur as a result of those commands.

Non-CMS error messages produced by the VSE ESERV program are described in the *Guide to the DOS/VSE Assembler*.

EXEC

Use the EXEC command to execute one or more CMS commands or exec control statements contained in a specified System Product Interpreter, CMS EXEC or EXEC 2 file.

Format

[EXec]	<i>fn</i> [<i>args...</i>]
--------	------------------------------

Operands

[EXec]

indicates that the EXEC command may be omitted if you are executing the exec procedure from the CMS command environment and have not issued the command SET IMPEX OFF.

fn

is the file name of a file containing one or more CMS commands and/or EXEC control statements to be executed. The file type of the file must be EXEC. The file can have either fixed- or variable-length records with a logical record length not exceeding 130 characters. A text editor or a user program can create exec files. Exec files that a CMS editor creates have, by default, variable-length, 80-character records.

args

are any arguments you wish to pass to the exec. The CMS EXEC processor assigns arguments to special variables &1 through &30 in the order in which they appear in the argument list. The EXEC 2 processor assigns arguments to special variables starting with special variable &1. With the System Product Interpreter and the EXEC 2 processors, the *number* of arguments is not limited. However, the number of bytes of data you can pass in the argument list is limited. The limit is the maximum number of bytes that can fit in a line: 130 bytes if the command is entered from a terminal, 255 bytes if the command is issued from a program. Arguments passed to the System Product Interpreter are handled differently than they are in EXEC or EXEC 2. See the *VM/SP System Product Interpreter Reference* for details.

For information about the System Product Interpreter, see the:

VM/SP System Product Interpreter User's Guide and the
VM/SP System Product Interpreter Reference.

See the *VM/SP EXEC 2 Reference* for information about EXEC 2.

Use the HELP facility to get complete descriptions of EXEC control statements, special variables, and built-in functions by issuing the command:

help exec menu

EXEC

Example

If the implied EXEC function is set to OFF (QUERY IMPEX to find out the setting), and if you want to execute your TPHONE EXEC with a nickname, you would enter enter:

```
exec tphone rick
```

Responses

The amount of information displayed during the execution of a CMS exec depends on the setting of the &CONTROL control statement. By default, &CONTROL displays all CMS commands, responses, and error messages. In addition, it displays nonzero return codes from CMS in the format:

```
+++ R(nnnnn) +++
```

where nnnnn is the return code from the CMS command.

The amount of information displayed during the execution of a System Product Interpreter file depends on whether tracing is set on. See the *VM/SP System Product Interpreter Reference* for details.

The amount of information displayed during the execution of an EXEC 2 file depends on the setting of the &TRACE control statement. See *VM/SP EXEC 2 Reference* for details.

Return codes for error messages from CP commands directly correspond to the message number. For example, if you issued a CP LINK command with an incorrect user ID, you receive error message DMKLNK053E *userid* not in CP directory. When issued from a CMS exec program, the same CP LINK command would have a return code of 53.

Messages and Return Codes

DMSEXC001E No filename specified

If the exec interpreter finds an error, it displays the message:

DMSEXT072E Error in EXEC file *fn*, line *nnn*: *message*

The possible errors, and the associated return codes, are:

Description	Return Code
FILE NOT FOUND	801
&SKIP OR &GOTO ERROR	802
BAD FILE FORMAT	803
TOO MANY ARGUMENTS	804
MAX DEPTH OF LOOP NESTING EXCEEDED	805
ERROR READING FILE	806
INVALID SYNTAX	807
INVALID FORM OF CONDITION	808
INVALID ASSIGNMENT	809
MISUSE OF SPECIAL VARIABLE	810
ERROR IN &ERROR ACTION	811
CONVERSION ERROR	812
TOO MANY TOKENS IN STATEMENT	813
MISUSE OF BUILT-IN FUNCTION	814
EOF FOUND IN LOOP	815
INVALID CONTROL WORD	816
EXEC ARITHMETIC UNDERFLOW	817
EXEC ARITHMETIC OVERFLOW	818

SPECIAL CHARACTER IN VARIABLE SYMBOL 819

If the EXEC 2 interpreter finds an error, it displays the message:

DMSEXE085E Error in *fn ft fm*, line *nnn* - *message*

The possible errors and the associated return codes are:

Description	Return Code
FILE NOT FOUND	10001
WRONG FILE FORMAT	10002
WORD TOO LONG	10003
STATEMENT TOO LONG	10004
INVALID CONTROL WORD	10005
LABEL NOT FOUND	10006
INVALID VARIABLE NAME	10007
INVALID FORM OF CONDITION	10008
INVALID ASSIGNMENT	10009
MISSING ARGUMENT	10010
INVALID ARGUMENT	10011
CONVERSION ERROR	10012
NUMERIC OVERFLOW	10013
INVALID FUNCTION NAME	10014
END OF FILE FOUND IN LOOP	10015
DIVISION BY ZERO	10016
INVALID LOOP CONDITION	10017
ERROR RETURN DURING &ERROR ACTION	10019
ASSIGNMENT TO UNSET ARGUMENT	10020
STATEMENT OUT OF CONTEXT	10021
INSUFFICIENT STORAGE AVAILABLE	10097
FILE READ ERROR <i>nnn</i>	10098
TRACE ERROR <i>nnn</i>	10099

DMSEXE175E Invalid EXEC command RC=10096

DMSEXE255T Insufficient storage for Exec interpreter RC=10000

For the System Product Interpreter, the possible errors and associated return codes are:

Description	Return Code
Program is unreadable	20003
Program interrupted	20004
Machine storage exhausted	20005
Unmatched <i>'/*'</i> or quote	20006
WHEN or OTHERWISE expected	20007
Unexpected THEN or ELSE	20008
Unexpected WHEN or OTHERWISE	20009
Unexpected or unmatched END	20010
Control Stack Full	20011
Clause > 500 characters	20012
Invalid character in data	20013
Incomplete DO/SELECT/IF	20014
Invalid Hex constant	20015
Label not found	20016
Unexpected PROCEDURE	20017
THEN expected	20018
String or symbol expected	20019
Symbol expected	20020

EXEC

Invalid data on end of clause	20021
Invalid character string	20022
Invalid TRACE request	20024
Invalid sub-keyword found	20025
Invalid whole number	20026
Invalid DO syntax	20027
Invalid LEAVE or ITERATE	20028
Environment name too long	20029
Name or String > 250 chars	20030
Name starts with numeric or “.”	20031
Invalid use of stem	20032
Invalid expression result	20033
Logical value not 0 or 1	20034
Invalid expression	20035
Unmatched “(” in expression	20036
Unexpected “,” or “)”	20037
Invalid template or pattern	20038
Evaluation stack overflow	20039
Incorrect call to routine	20040
Bad arithmetic conversion	20041
Arithmetic overflow/underflow	20042
Routine not found	20043
Function did not return data	20044
No data on function RETURN	20045
Failure in system service	20048
Interpreter failure	20049

DMSREX255T Insufficient storage for Exec interpreter RC = 20096 [RC = 10096]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813

EXECDROP

Use the EXECDROP command to remove the specified exec(s) or System Product Editor macro(s) from storage or to discontinue use of the specified exec(s) or editor macro(s) in a CMS installation saved segment.

Format

EXECDrop EXDrop	$\left\{ \begin{array}{c} \textit{execname} \\ * \end{array} \right\} \left[\begin{array}{c} \textit{exectype} \\ * \end{array} \right] [(\textit{options...})]$
	Options: $\left[\begin{array}{c} \text{User} \\ \text{SYstem} \\ \text{SHared} \end{array} \right]$

Operands

execname

is the name of the storage-resident exec(s) to be purged. If an asterisk (*) is coded in this field, all execnames are used.

exectype

is the type of the storage-resident exec(s) to be purged. If an asterisk (*) is coded in this field, all exectypes are used. The default is an asterisk (*).

*

specified alone indicates that all storage-resident execs are purged.

Options

User

indicates that the storage for the exec(s) was allocated from user free storage. Only the execs that satisfy the execname and/or exectype qualifications and that also have the USER attribute are dropped. If neither USER, SYSTEM, nor SHARED is specified, then all execs with the USER and SYSTEM attributes that satisfy the execname and/or exectype qualifications are dropped.

SYstem

indicates that the storage for the exec(s) was allocated from nucleus free storage. Only the execs that satisfy the execname and/or exectype qualifications and that also have the SYSTEM attribute are dropped. If neither USER, SYSTEM, nor SHARED is specified, then all execs with the USER and SYSTEM attributes that satisfy the execname and/or exectype qualifications are dropped.

SHared

indicates that the exec(s) was located in a CMS installation saved segment. Only the execs that satisfy the execname and/or exectype qualifications and that also have the SHARED attribute are dropped for the duration of your CMS session. If neither USER, SYSTEM, or SHARED is specified, then all execs with the USER and SYSTEM attribute that satisfy the execname and/or exectype qualifications are dropped.

EXECDROP

SHARED execs can only be dropped when SET INSTSEG is ON. Once a SHARED exec is dropped, you can only reload it when you IPL CMS. To discontinue use of all SHARED execs temporarily, use the SET INSTSEG OFF command.

Examples

To drop all storage-resident execs, specify:

```
execdrop *
```

To purge all storage-resident execs with exectypes of XEDIT that were loaded into user free storage using the EXECLOAD command, specify the following:

```
execdrop * xedit (user
```

Messages and Return Codes

DMSEXD003E Invalid option: *option* [RC=24]
DMSEXD042E No execid specified [RC=24]
DMSEXD070E Invalid parameter *parameter* [RC=24]
DMSEXD109S Virtual storage capacity exceeded [RC=104]
DMSEXD415E Invalid character *char* in execid *execname exectype* [RC=20]
DMSEXD416W There are no *execname exectype* {system|[or]user|[or]shared}
EXECs storage resident [RC=28]
DMSEXD418W Drop pending for *execname exectype* [RC=4]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

EXECIO

Use the EXECIO command to:

- Read lines from a disk, directory, or virtual reader to the program stack or a variable.
- Write lines from the program stack or a variable to a CMS file or to a virtual spool device (punch or printer).
- Cause execution of CP commands and recover resulting output.

In some cases output data to be written may be supplied directly on the EXECIO command line.

The information immediately following is reference level information about EXECIO format and operands. Following this reference information you can find extended descriptive and use information. If you are not familiar with EXECIO, you should review the complete command description before attempting to use it. Also, to get full benefit from EXECIO you should be familiar with use of execs under the System Product Interpreter or EXEC 2 (Refer to the *VM/SP System Product Interpreter Reference* or to the *VM/SP EXEC 2 Reference*).

In the following descriptions, “relative line number” means the number of lines processed to satisfy an EXECIO operation; “absolute line number” means the number of the line relative to the top of the file.

Format

EXECIO	<pre> { lines } { DISKR fn ft [fm [linenum]] [(FINIs) options [a] [b]] [] } { * } { CARD [(options [a] [b]] []) CP [(options [a] [b] [d] [e]] []) DISKW fn ft fm [linenum [recfm [lrecl]]] [(FINIs) options [b] [c] [d]] [] } PUNCH[(options [b] [c] [d]] []) PRINT [([CC {code }] options [b] [c] [d]] []) [DATA]] EMSG [(options [b] [c] [d]] [])] </pre> <p>Option formats:</p> <p>(a)</p> <pre> [FInd /chars /] [Zone { n1 n2 }] [LIFO] [SKip] [LOcate /chars /] [1 *] [FIFO] [Avoid /chars /] </pre> <p>(b)</p> <pre> [Margins { n1 n2 }] [STRIP] [NOTYPE] [STEm xxxxn] [1 *] [VAR xxxx] </pre> <p>(c)</p> <pre> [CAse { U }] [M] </pre> <p>(d)</p> <pre> [STring xxx...] </pre> <p>(e)</p> <pre> [BUfFer length] </pre>
---------------	---

Note: Parsing of the EXECIO command differs from that of other CMS commands in that it involves handling of strings that may contain embedded blanks, parentheses, other special characters, and words of more than eight characters. Therefore, if a right parenthesis is used to mark the end of an EXECIO option, it must be preceded by at least one blank character. A right parenthesis cannot be used to mark the end of the STRING option.

Operands

lines

is the number of source lines processed. This can be any non-negative integer. With the VAR option, the number of lines must be 1. An asterisk (*) indicates that the operation is to terminate when:

1. a null (0-length) line is read during an *output* operation;
2. an end-of-file condition is detected during an *input* operation.

Specification of *, together with the STRING option, is valid only with the CP operand. Using the * and STRING combination with any other operand causes an error message to be issued. Also the combination of the * and the VAR options is not allowed. If *lines* is specified as zero (0), no I/O operation is performed other than FINIS, when it is specified as an option.

DISKR

is used to read a specified number of lines from the CMS file “fn ft [fm]” to the program stack FIFO (first-in first-out) or to an EXEC 2 or System Product Interpreter variable if the STEM or VAR options are specified.

CARD

is used to read a specified number of lines from the virtual reader to the program stack (FIFO) or to an EXEC 2 or System Product Interpreter variable if the STEM or VAR options are specified.

CP

causes output resulting from a CP command to be placed on the program stack (FIFO) or to an EXEC 2 or System Product Interpreter variable if the STEM or VAR options are specified. To obtain the reply from a CP command, specify the *lines* operand as an asterisk (*). If you want to issue a command to CP, suppressing messages and obtaining only the return code, specify the *lines* operand as zero (0). You may specify which CP command is to be issued via:

1. the STRING option on the EXECIO command line;
2. the next line from the program stack.

Keep in mind that all characters of CP commands must be in upper case.

DISKW

is used to write a specified number of lines from the program stack or from an EXEC 2 or System Product Interpreter variable if the STEM or VAR options are specified to a new or existing CMS file “fn ft fm.”

Inserting a line into a variable length CMS file can cause truncation of the portion of the file following the inserted line. See the extended DISKW operand description.

recfm

lrecl

define the record format and record length for any *new* file created as a result of a DISKW operation. The default value for *recfm* is V (variable), in which case “*lrecl*” has no meaning. If you specify F (fixed) for *recfm*, the default *lrecl* value is 80. The maximum *lrecl* value that may be specified is 255 unless the VAR or STEM option is used to bypass the use of the program stack, in which case the maximum is that which is defined for the type of file in use (800 byte or EDF file). If *recfm* or *lrecl* is specified for the DISKW operation, then a *linenum* value must be specified explicitly.

PUNCH

is used to transfer a specified number of lines from the program stack or from an EXEC 2 or System Product Interpreter variable if the STEM or VAR options are specified to the virtual punch.

PRINT

is used to transfer a specified number of lines from the program stack or from an EXEC 2 or System Product Interpreter variable if the STEM or VAR options are specified, to the virtual printer using the PRINTL macro.

CC

is used with the PRINT operand to specify carriage control for each line transferred to the virtual printer. Using the CC operand, you can supply carriage control code explicitly, or by specifying DATA, indicate that the carriage control character is the first byte of each line.

Note: If you wish to print from a program stack, and your virtual printer has a device type that reflects channel code 9 or 12 sensing back to the user, see the section of EXECIO called CC operand for additional information.

code

is the character (ASA or machine code) that defines carriage control. A blank code (the default value) cannot be specified on the command line.

DATA

specifies that the first byte of each line sent to the virtual printer is a carriage control character.

EMSG

causes a message to be displayed. The text of the message may be:

1. the character string specified on the STRING option;
2. the next available line(s) from the program stack;
3. the information from a STEM or VAR variable.

Messages are edited according to the current CP SET EMSG settings.

fn

is the file name of the file.

ft

is the file type of the file.

fm

is the file mode of the file. When file mode is specified, that disk or directory and its extensions are searched. If file mode is not specified, or is specified as an asterisk (*), all accessed disks and directories are searched for the specified file.

linenum

is the absolute line number within the specified file where a DISKR or DISKW operation is to begin. If a value is zero or not specified for newly opened files, reading begins at the first line and writing begins at the last line. For other files, reading or writing begins at the line following the one at which the previous operation ended. Because EXEC processors manipulate execs that are currently executing, a read or write to a currently running exec should explicitly specify the linenum operand. Failure to do so may cause the first line to be read or written each time. If recfm or lrecl is specified for the DISKW operation, then a linenum value must be specified explicitly.

FINIs

causes the specified file to be closed following completion of a DISKR or DISKW operation.

Option A**FInd**

is used to write to the program stack LIFO (last-in first-out) or FIFO (first-in first-out) to an EXEC 2 or System Product Interpreter array.

1. the contents of that line;
2. the line number of the first occurrence of a line (or zone portion of that line) that *begins with* the characters specified between delimiters. For DISKR operations both the relative and absolute line numbers are written. Otherwise, only the relative line number is written.

If you wish to *search* only a portion of each line, use the ZONE option, explained below. If you wish to *write* only a portion of any line matching the search argument to the program stack or a variable, use the MARGINS option, also explained below.

LOCate

is like the FIND option explained above, except that the object characters may occur any place within a line (or zone portion of that line).

Avoid

is like the LOCATE option explained above, except that the search is for a line (or zone portion of that line) that *does not* contain the specified characters.

Zone

is used to restrict the portion of the input lines searched as a result of the FIND, LOCATE, or AVOID options. The search is between columns n1 and n2 (inclusive), if specified. The default values are column 1 through the end of the line (*). The limits of values that may be specified for n1 or n2 are 1 through $2^{31}-1$.

LIFO**FIFO**

defines the order in which lines are written to the program stack. Generally, the default order is FIFO (first-in first-out). The exceptions are operations that put line numbers on the program stack as a result of a search operation (FIND, LOCATE, or AVOID). These operations default to LIFO (last-in first-out). The use of VAR or STEM is not valid with this option.

SKip

allows a read function (DISKR, CARD, CP) to occur without writing any information to the program stack.

Option B**Margins**

specifies that only a portion (columns n1 through n2 inclusive) of affected lines is to be processed (from the stack or a variable). The default values are column 1 through the end of each line (*). The limits of values that may be specified for n1 or n2 are 1 through $2^{31}-1$.

STRIP

specifies that trailing blank characters are to be removed from any output lines or lines returned.

EXECIO

NOTYPE

suppresses the display of message DMSEIO632E at the virtual console.

STEM *xxxxn*

indicates that the variables *xxxxn* are to be used to supply input data for output-type operations (PUNCH, PRINT, EMSG, and DISKW) or that variables *xxxxn* are the destination for output for the input-type operations (CARD, DISKR, and CP). The variables used are a concatenation of the specified stem and a number that corresponds to the number of lines of output. The special variable *xxxx0* is set to the number of lines of information returned for the input-type operations. STEM can be used with the STRING operation only with the CP operation. The maximum length variable name with the STEM option is 240 bytes.

VAR *xxxx*

indicates that the variable *xxxx* is to be used to supply input data for output-type operations (PUNCH, PRINT, EMSG, and DISKW) or that variable *xxxx* is the destination for output for the input-type operations (CARD, DISKR, and CP). If VAR is specified, then the number of lines must be 1. VAR cannot be used with the FIFO or LIFO options, nor with the FIND, AVOID, and LOCATE options because they cause more than one line of output to be written. VAR can be used with the STRING option only with the CP operation. The maximum length variable name is 250 characters.

Option C

CAse

causes data read from the program stack, from a STEM, or from a variable to be:

1. translated to uppercase if U is specified;
2. not translated (mixed) if M is specified.

M (mixed) is the default value.

Option D

STring

is used to supply output data explicitly on the EXECIO command line. Any characters following the STRING keyword are treated as string data, not additional EXECIO operands. Therefore, STRING, if specified, must be the final option on the command line.

Option E

BUFfer *length*

specifies the length, in characters (bytes), of the CP command response expected from a CP operation. The limits of values that may be specified for *length* are 1 through $2^{31}-1$. If this option is not specified, up to 8192 characters of the response are returned.

Extended Descriptions and Use Information

General comments:

EXECIO commands are normally issued as statements from System Product Interpreter or EXEC 2 EXECs. Because some EXECIO option values can exceed eight characters, an extended parameter list is required for EXECIO execution when these options are specified. Otherwise, an error message results. Both the System Product Interpreter and EXEC 2 supply a tokenized parameter list and, if necessary, an extended parameter list.

You should keep in mind that when a CMS operation completes and the READY message (Ready;) displays, CMS closes all files. Any subsequent EXECIO read operation will begin at file line one unless a “linenum” value is specified. Any subsequent EXECIO write operation will begin at the end of the file unless a “linenum” value is specified. Therefore, when possible, it is a good idea to specify a “linenum” value on the EXECIO command line.

For write operations, data to be written is normally taken from the program stack. However, data to be written may be supplied via the STRING option or via the VAR or STEM options (in all cases the exec in question must be a System Product Interpreter exec or an EXEC 2 exec).

Program stack:

The program stack is a buffer area, expanded as necessary from available free storage. Data flow into and out of the program stack is:

1. normally FIFO (first-in first-out) for read or write operations;
2. LIFO (last-in first-out) for read options, such as FIND or LOCATE, that result in a line number being stacked.

A successful search (LOCATE, FIND, etc.) operation results in two lines being written (LIFO) to the program stack:

1. the contents of the line that satisfied the search argument;
2. the relative line number (number of lines read to obtain a match for the search argument), and for a DISKR operation only, the absolute line number (position from the top of the file).

Stacked line number values may be used on subsequent EXECIO operations for *lines* or *linenum* operands.

The CMS SENTRIES command results in a return code equal to the number of lines in the program stack. Thus, the difference between SENTRIES return codes, one before and one after an EXECIO operation, is the number of lines stacked as a result of that operation.

If the exec is coded in the REXX language, then the QUEUED() built-in function can be used to return the same information as the CMS SENTRIES command. Refer to the *VM/SP System Product Interpreter Reference* for information on the QUEUED() built-in function.

Note: During a print operation, if carriage control (CC) is used, and you have a virtual printer with a device type that uses channel code sensing, see the operand section, *CC operand*, for more information.

Setting Variables Directly:

The VAR and STEM options allow EXEC 2 and System Product Interpreter variable(s) to be set directly, bypassing the use of the stack. Data flow in and out of the variable(s) is:

- always FIFO (first-in first-out) for read or write operations using STEM variables.

- FIFO for read options, such as FIND or LOCATE, that result in a line number being returned.

A successful search (LOCATE, FIND, etc.) operation results in two lines being assigned (FIFO) to EXEC variables if the STEM option has been used:

1. the contents of the line that satisfies the search argument;
2. the relative line number (number of lines read to obtain a match for the search argument), and for a DISKR operation only, the absolute line number (position from the top of the file).

Returned line number values may be used on subsequent EXECIO operations for “lines” or “linenum” operands.

If the STEM option was specified, then variable xxxx0 contains the number of lines of data returned.

For VAR and STEM operations called from the System Product Editor, the maximum that EXECIO processes is $2^{31}-1$; if called from an EXEC2 program, the maximum value is 255. If the command returns a non-zero return code, the variables specified by the command will be undefined and cannot be used.

Closing files and virtual devices:

EXECIO (DISKR or DISKW) operations do not close referenced files when the operation terminates unless the FINIS operand is specified on the command line. (Some CMS commands, such as ERASE and STATE, close the referenced file, then execute.) If you choose, you may close these files manually by invoking the CMS FINIS command. For files in an SFS directory, changes are not committed when there are files or directories open on the same work unit. Using the FINIS option (or the FINIS command) to close any open files will commit your changes. There is considerable system overhead associated with the execution of FINIS. Therefore, if multiple references are to be made to a given file, it should be closed only when necessary.

If successive EXECIO commands reference a particular internal area of a CMS file, it is probably more efficient to let the file remain open until the last of these commands is issued. If so, each operation begins at the file line following the last line processed. This eliminates much of the need for calculating the “linenum” value.

EXECIO does not close virtual spool devices. Therefore, to cause any spooled EXECIO output to be processed you must close the corresponding device. For example:

```
CP CLOSE PRINTER
```

If an input spool file is read with the EXECIO CARD operation and the read is not completed (that is, the virtual machine does not get a last-card indication), you must issue a CP CLOSE READER command to be able to read that file again (or to read any other file). The file is purged unless you specify HOLD when you close a reader file. Refer to the CP CLOSE command in the *VM/SP CP General User Command Reference*.

If you have specified the PRINT or PUNCH operand and you try to write a line that is longer than the virtual device (PRINTER or PUNCH) allows, you will get error message DMSEIO632E.

lines operand:

For a DISKW, PUNCH, PRINT, or EMSG operation (if the STEM option had not been specified), if the *lines* operand exceeds the number of lines on the program stack, reading continues through the terminal input buffer. If the *lines* operand is still not satisfied, a VM READ is issued to the terminal. At that point, you must enter the balance of the lines (the number specified in the *lines* operand) from the terminal. Entering a blank character (null line) does not terminate the EXECIO operation; it writes a blank character to the output device. When the *lines* operand has been satisfied, the exec from which EXECIO was issued continues to execute.

If * (to end of file) is specified for *lines* on an output operation, and you want the operation to terminate at any given line in the program stack or a STEM array, you must make sure that line is null. Reading a null line terminates any of the four output operations if * is specified for the *lines* operand.

For input operations (DISKR, CARD, and CP), the number of lines written to the program stack or the STEM array does not necessarily equal the number specified by *lines*. For example, an end-of-file or a satisfied search condition terminates a read operation, even if the specified number of lines has not been written to the program stack or the STEM array. When a search argument (FIND, LOCATE, AVOID option) is satisfied, and no SKIP option is specified, and the default stacking order (LIFO) is used, the line at the top (first line out) of the stack or the STEM array contains the number of operations required to satisfy the search. The next line contains the line that satisfied the search.

If the search argument (FIND, LOCATE, AVOID option) is not satisfied, a return code of 3 is given, even if EOF occurs before the specified number of lines has been read. A return code of 3 is also given if * is specified for *lines* on a read operation, and the search argument is not satisfied.

When a number greater than 0 is specified for *lines* with output operation CP, and the number of lines written to the program stack, stem array, or variable name is not equal to the number specified by *lines*, a return code of 2 is given.

When * is specified for *lines* on a read operation, the operation is terminated at end-of-file. A return code of 0 is given because the * is an explicit request to read to end-of-file.

When a search argument (FIND, LOCATE, and AVOID options) is not satisfied and an end-of-file situation occurs for the EXECIO CARD operation, the reader file is purged unless a CP SPOOL READER HOLD was previously specified. For more information on how spool file are processed, refer to the CP CLOSE and CP SPOOL commands in the *VM/SP CP General User Command Reference*.

DISKR operation:

The first line read on a DISKR operation may be:

1. the first line of the specified file;
2. specified using the "linenum" operand;
3. determined by the results of a previous operation.

EXECIO

The DISKR operation may be used to simply read a specified number of lines from a specified file and write them to the program stack or a variable. For example, suppose file MYFILE DATA contains:

```
The number one color is red
The number two color is yellow
The number three color is green
The number four color is blue
The number five color is black
```

The command:

```
EXECIO 2 DISKR MYFILE DATA * 1
```

writes to the program stack (FIFO) two lines beginning with line one, like this:

```
| The number one color is red    |<-next line read
| The number two color is yellow |
|                               |
/                               /
```

However, a little more complex version of this command:

```
EXECIO 2 DISKR MYFILE DATA * 3 (LIFO MARGINS 5 14
```

would have resulted in this program stack:

```
| number fou                      |<-next line read
| number thr                      |
|                               |
/                               /
```

Note the use of * as a file mode operand on the command lines just above to serve as a place holder. The command:

```
EXECIO 2 DISKR MYFILE DATA * 1 (STEM X.
```

assigns to variables X.1 and X.2 one line each, beginning with line one, like this:

```
| The number one color is red    | X.1
| The number two color is yellow | X.2
|                               |
/                               /
```

The X.0 variable contains the number of lines (in this example, 2).

When a line satisfies the LOCATE, FIND, or AVOID option for a DISKR operation, EXECIO writes that line to the program stack (LIFO) or a variable (FIFO) and in an additional stack line or variable, writes the relative (number of lines read to satisfy the search) and absolute (position from the top of the file) line numbers.

CP operand:

When a search argument is required, the CP operand uses the FIND, LOCATE, and AVOID options to process output resulting from the associated CP command. Each line that satisfies the search criteria is written to the program stack or a variable. When data exceeds 8192 characters, it is truncated and an error code is returned. If you specify the BUFFER option, data is truncated after the number of characters specified in *length* or 8192 is reached, whichever is greater. The number of read operations required to match the search argument is written to the next stack line.

If you do not supply the CP command to be issued via the STRING option, the next line in the program stack is treated as that command. If there are no lines in the

program stack, the next line in the console input buffer is treated as the CP command. If there are no lines in the console input buffer, then a VM READ is issued to the terminal. A null line terminates the operation.

The responses from the CP command are treated as input. If CP SET IMSG is set OFF, no response is issued by some CP commands. This may result in a return code of 2, if a number other than zero (0) is specified for *lines*. The return code of 2 indicates that the end of the input file was reached before the specified number of lines could be read. This will not occur if you specify the *lines* operand as an asterisk (*). For more information regarding which CP commands are affected by the setting of IMSG refer to the CP SET command in the *VM/SP CP General User Command Reference*.

Keep in mind that all characters of CP commands must be uppercase.

ZONE and MARGINS options do not affect the reading of the CP command; however, they do affect the portions of the lines processed as a result of the command execution.

DISKW operand:

The DISKW operand causes the next lines from the program stack to be written to a CMS file. The point at which writing begins in an existing file on a DISKW operation may:

1. follow the last file line (default "linenum" when writing to a newly opened file, for example);
2. be specified using the "linenum" operand;
3. be determined by the results of a previous operation.

For example, suppose you want to write 10 lines from the program stack to the end of an existing file, BUCKET STACK A, on your disk or directory accessed as A. Your exec file statement to do this would be:

```
EXECIO 10 DISKW BUCKET STACK A
```

Now, take a slightly more complex requirement. Using stack lines down to the first null line, create a new file, BASKET STAX A, then close the file after it is written. Also, make the file fixed length format with a record length of 60. The EXECIO command to do this is:

```
EXECIO * DISKW BASKET STAX A 1 F 60 (FINIS
```

A word of caution about using the linenum operand to insert lines in the middle of CMS variable length files. Because of the way CMS handles these files, any variable length line inserted must be equal in length to the line it displaces. Otherwise, for minidisks formatted in:

1. 512-, 1K-, 2K-, or 4K-byte blocks, all file lines following the one inserted are truncated;
2. 800-byte blocks, the file remains unchanged and CMS issues message 105S (nn = 15).

Also for files in an SFS directory, all lines following the one inserted are truncated.

For example, assume a disk format in 2K-byte blocks. The variable format file WORDS LEARNING A is:

A is for apple
C is for cake
C is for candy
D is for dog

execution of:

EXECIO 1 DISKW WORDS LEARNING A 2 (STRING B is for butterfly

produces a file that contains only:

A is for apple
B is for butterfly

Because "B is for butterfly" contains more characters than the line it writes over, "C is for cake," all lines following it are truncated. However, slightly modifying the command to:

EXECIO 1 DISKW WORDS LEARNING A 2 (STRING B is for baby

results in:

A is for apple
B is for baby
C is for candy
D is for dog

To prevent truncation when inserting records in a variable-length file, you can use fixed-format files.

recfm

lrecl operands:

The default value for *recfm* is V (variable), in which case "lrecl" has no meaning. If you specify F (fixed) for *recfm*, the default *lrecl* value is 80. The maximum *lrecl* value that you may specify is 255 unless the VAR or STEM option is used to bypass the use of the program stack, in which case the maximum is that which is defined for the type of file in use (800 byte or EDF file).

When lines are written to an existing file, the record format and record length of that file apply. Specifying *recfm* or *lrecl* values on the EXECIO command line that conflict with those of the existing file causes an error message to be issued.

CC operand:

When you specify CC together with the DATA operand, be sure the first character of each line to be sent to the virtual printer may be removed and interpreted as carriage control for that line.

You may use ASA or machine code characters with the CC operand to specify carriage control. For example, CC 0 causes space two lines before printing. You can find information about these codes under the PRINTL macro description in *VM/SP Application Development Reference for CMS*. If your virtual printer is a device type which reflects channel code sensing back to you, the sensing of channel code 12 or 9 results in a return code of 2 or 3 from PRINTL, which EXECIO reflects as return code 102 or 103. For these type conditions, the following options are available for you to handle recovery:

- Code the application to examine the return code from EXECIO and retry the print operation if a channel code 12 or 9 has been detected.

- Redefine the virtual printer to a device type that does not reflect channel 12 or 9 sensing.
- Redefine the FCB for the printer to eliminate the channel code 12 or 9.

EMSG operand:

Lines to be displayed by EMSG should have the format:

xxxmmnnns

where:

xxxmmm is the issuing module name

nnn is the message number

s indicates the message type (E - error, I - informational, W - warning etc.)

The current settings of the CP SET EMSG command control the displayed lines. These settings, combined with message length, can cause messages to be abbreviated or not displayed at all.

linenum operand:

When a linenum value (default 0) is not specified on the EXECIO command line, the number of the next file line available for reading or writing depends on results of previous operations that referenced that file. For example, consider the two EXECIO DISKR operations just below. By looking at the first of these commands you can see:

1. Four lines are to be read from MYFILE DATA, starting at line 1;
2. Because FINIS is not specified on the command line, MYFILE DATA remains open after the first read operation. Because the first command reads 4 lines, the subsequent read operation will begin at line 5.

```

      .
EXECIO 4 DISKR MYFILE DATA * 1
      .
      .
EXECIO 3 DISKR MYFILE DATA (FINIS
      .

```

Because the second EXECIO command specifies no linenum operand, reading of the specified 3 lines begins at line 5.

Two situations that would cause the second EXECIO command to not begin execution at line 5 are:

1. a program other than EXECIO accessing MYFILE DATA after the first and before the second EXECIO command is executed;
2. a CMS operation completing such that the CMS READY message (Ready;) is displayed. In that case CMS closes associated files. Therefore, subsequent operations using these files would begin at line 1.

The FINIS operand causes MYFILE DATA to close. Therefore, any subsequent DISKR operation using a default linenum value would begin reading at line 1.

FIND
LOCATE
AVOID options:

The delimiter pair for the specified character string need not be // . They may be any character not included in the string. For example:

```
EXECIO * DISKR MYFILE DATES (LOCATE $12/25/81$
```

FIFO
LIFO options:

Most EXECIO operations that write to the program stack default to FIFO, first line written to the stack will be the first read out. The exceptions (LIFO) are operations involving a search (LOCATE, FIND, and AVOID options). These operations result in the relative line number (number of lines read to satisfy the search) being stacked. For DISKR operations the absolute line number (position from the top of file) is also stacked on the same line. It is necessary to have these numbers at the top of the stack so that they are immediately accessible to a subsequent EXECIO command.

SKIP option:

On EXECIO read operations the SKIP operand prevents input lines from being written to the program stack or a variable. For example, you might want to put on the program stack all lines of MYFILE DATA that follow the line containing "4120 Rock Road." First, to search through the file for the line after which reading to the program stack is to begin, issue:

```
EXECIO * DISKR MYFILE DATA * 1 (LOCATE /4120 Rock Road/ SKIP
```

The SKIP option prevents the line being searched for, together with the line number, from being written to the program stack. Then, to write to the program stack the next line through the end of file, issue:

```
EXECIO * DISKR MYFILE DATA
```

Keep in mind that accessing MYFILE DATA by another program or causing a CMS READY message to be displayed prior to issuing the second EXECIO command would change the point at which the second command begins reading. When possible, you should specify the linenum operand explicitly.

Another use of the SKIP option might be the execution of a CP command via the CP operand to obtain a return code without displaying the resulting messages or writing them to the program stack or a variable. For example:

```
EXECIO * CP (SKIP STRING Q userid
```

The user ID must be uppercase.

As an alternative, specifying 0 for the *lines* operand value with the CP operand also causes results not to be displayed or written to the program stack.

STEM option:

On EXECIO operations, the STEM option allows an array of EXEC 2 or System Product Interpreter variables to be set directly, bypassing the use of the stack. For example, if you want the first 3 lines of MYFILE DATA to be assigned to System Product Interpreter variables X.1, X.2, and X.3, issue:

```
'EXECIO 3 DISKR MYFILE DATA * 1 (STEM X.'
```

Variable X.1 now contains the first line of MYFILE DATA, variable X.2 contains the second line, and variable X.3 contains the third line. Variable X.0 contains the number of lines (in this example, 3).

For System Product Interpreter variables, the variable name should be enclosed in quotes and must be in uppercase. For EXEC 2 variables, you omit the '&', quotes are not necessary, and mixed case may be used.

Note: Doing a DISKR operation using the STEM option from an EXEC 2 exec may cause truncation to 255 bytes and a return code of 1.

For the STEM option, the maximum length variable name is 240 bytes.

VAR option:

On EXECIO operations, the VAR option allows an EXEC 2 or System Product Interpreter variable to be set directly, bypassing the use of the stack. For example, if you want the second line of MYFILE DATA to be assigned to an EXEC 2 variable named X, issue:

```
EXECIO 1 DISKR MYFILE DATA * 2 (VAR X
```

Variable X now contains the second line of MYFILE DATA.

For System Product Interpreter variables, the variable name should be enclosed in quotes and must be in uppercase. For EXEC 2 variables, you omit the '&', quotes are not necessary, and mixed case may be used.

Note: Doing a DISKR operation using the VAR option from an EXEC 2 exec may cause truncation to 255 bytes and a return code of 1.

For the VAR option, the maximum length variable name is 250 bytes. See the *VM/SP CMS User's Guide* for an example of using EXECIO within System Product Interpreter programs.

Messages and Return Codes

DMSEIO621E	Bad Plist: Device and lines arguments are required [RC = 24]
DMSEIO621E	Bad Plist: Invalid value <i>value</i> for number of lines [RC = 24]
DMSEIO621E	Bad Plist: Missing DEVICE argument [RC = 24]
DMSEIO621E	Bad Plist: Invalid DEVICE argument <i>argument</i> [RC = 24]
DMSEIO621E	Bad Plist: Disk <i>argument</i> argument is missing [RC = 24]
DMSEIO621E	Bad Plist: Invalid character in file identifier [RC = 24]
DMSEIO621E	Bad Plist: Invalid value <i>value</i> for disk file line number [RC = 24]
DMSEIO621E	Bad Plist: Disk filemode required for DISKW [RC = 24]
DMSEIO621E	Bad Plist: Invalid record format <i>recfm</i> -- Must be either F or V [RC = 24]
DMSEIO621E	Bad Plist: Invalid record length argument <i>lrel</i> [RC = 24]
DMSEIO621E	Bad Plist: File format specified <i>recfm</i> does not agree with existing file format <i>recfm</i> [RC = 24]
DMSEIO621E	Bad Plist: File <i>lrecl</i> specified <i>lrecl</i> does not agree with existing file <i>lrecl</i> [RC = 24]
DMSEIO621E	Bad Plist: Invalid positional argument <i>argument</i> [RC = 24]
DMSEIO621E	Bad Plist: EXECIO options only allowed with extended plist [RC = 24]
DMSEIO621E	Bad Plist: Unknown option name <i>name</i> [RC = 24]
DMSEIO621E	Bad Plist: Value missing after <i>option</i> option [RC = 24]
DMSEIO621E	Bad Plist: Value <i>value</i> not valid for <i>option</i> option [RC = 24]

EXECIO

DMSEIO621E Bad Plist: *option* option is not valid with *option* option [RC=24]
 DMSEIO621E Bad Plist: *option* option not valid with *operation* operation [RC=24]
 DMSEIO621E Bad Plist: STRING option with LINES=* is valid only for CP operation [RC=24]
 DMSEIO621E Bad Plist: VAR option with LINES>1 is invalid [RC=24]
 DMSEIO621E Bad Plist: Invalid mode *mode* [RC=24]
 DMSEIO621E Bad Plist: Invalid EXEC variable name [RC=24]
 DMSEIO621E Bad Plist: NAMEFIND must be invoked as a nucleus extension [RC=24]
 DMSEIO621E Bad Plist: QUERY must be invoked as a nucleus extension [RC=24]
 DMSEIO622E Insufficient free storage for EXECIO [RC=*rc*]
 DMSEIO631E *function* can only be executed from an EXEC-2 or REXX EXEC [RC=*rc*]
 DMSEIO632E I/O error in EXECIO: rc=*nnnn* from *command* command [RC=*rc*]
 DMSERD257T Internal system error at address *addr* (offset *offset*)
 DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
 DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814
Errors in printing a file	406

Return Code Definitions:

Return Code	Definition
0	Finished correctly
1	Truncated
2	EOF before specified number of lines were read
3	Count ran out without successful pattern match
24	Bad PLIST
31	Error caused a rollback of a shared file(s)
41	Insufficient free storage to load EXECIO
55	APPC/VM communication error
70	SFS sharing conflict
76	SFS authorization error
99	Insufficient virtual storage for SFS file pool
1nn	100 + return code from I/O operation (if nonzero)
2008	Variable name supplied on STEM or VAR option was invalid.
x1nnn	1000 + return code from CP command (if nonzero), where x is 0, 1, 2, or 3, as described above
1xnnnn	10000 + return code from CP command (if nonzero), where x is 0, 1, 2, or 3, as described above

Note: For information on the EXECCOMM function and the associated return codes, refer to the *VM/SP System Product Interpreter Reference*.

EXECLOAD

Use the EXECLOAD command to load an exec or System Product Editor macro into storage and prepare it for execution.

Format

EXECLoad EXLoad	$\{fn\} \{ft\} [fm \ [execname \ [exectype]]] \ [(\text{options...})]$ <u>Options:</u> [<u>User</u>] [Push] [<u>S</u> System]
----------------------------------	--

Operands

fn

is the file name of the exec to be loaded.

ft

is the file type of the exec to be loaded.

fm

is the file mode of the exec to be loaded. The default for file mode is an asterisk (*), which means the first file in the search order that satisfies the file name and file type qualifications is loaded. In order to assign an execname and exectype, you must also specify the file name, file type and file mode.

execname

is the name to be assigned to the loaded exec. The default is '=', which means the exec's present file name is to be used.

exectype

is the type to be assigned to the loaded exec. The default is '=', which means the exec's present file type is to be used.

Options

User

specifies that the storage for the loaded exec is allocated from user free storage. This is the default.

SSystem

specifies that the storage for the loaded exec is allocated from nucleus free storage.

Push

specifies that the exec is loaded whether an exec by the same name already exists in storage. This loaded exec does not replace the existing exec. Subsequent invocation of this execname and exectype executes the most recently loaded version. Also, a subsequent EXECDROP of this *execname* and *exectype* drops the most recently loaded version.

EXECLOAD

Usage Notes

1. The file name and file type can each be from one to eight characters. The valid characters are A-Z, a-z, 0-9, \$, #, @, +, - (hyphen), : (colon), and _ (underscore). The execname and exectype of the execid may also be from one to eight characters. However, the execname and exectype are not limited to the file name and file type character set. The only characters that are invalid within an execname or exectype are =, *, (,), and X'FF'.
2. If SET INSTSEG is ON and you attempt to load an exec into storage with the same execname and exectype as a SHARED exec, you must specify the PUSH option.
3. To list the execs in storage and in a CMS installation saved segment, use the EXECMAP command. To remove an exec from storage or to discontinue use of an exec in a CMS installation saved segment, use the EXECDROP command. Use the SET INSTSEG OFF command to discontinue use of all SHARED execs temporarily. To determine the status of a specific exec, use the EXECSTAT command.
4. To list the execs that are currently storage-resident, use the EXECMAP command. To purge a storage-resident exec, use the EXECDROP command. To determine the status of a specific exec, use the EXECSTAT command.

Examples

The following command:

```
execload tphone exec a (user
```

loads the TPHONE EXEC from your disk or directory accessed as A into user free storage and assigns it the same name.

Specifying the following:

```
execload tphone exec a = xedit (system
```

loads the TPHONE EXEC A into nucleus free storage and assigns to it the name TPHONE XEDIT.

Messages and Return Codes

DMSEXL003E	Invalid option: <i>option</i> [RC = 24]
DMSEXL042E	No fileid specified [RC = 24]
DMSEXL054E	Incomplete fileid specified [RC = 24]
DMSEXL062E	Invalid character * in fileid [RC = 20]
DMSEXL070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSEXL104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]
DMSEXL109S	Virtual storage capacity exceeded [RC = 104]
DMSEXL414E	Execid <i>execname exectype</i> already in storage [RC = 1]
DMSEXL415E	Invalid character <i>char</i> in execid <i>execname exectype</i> [RC = 20]
DMSEXL417E	Only EXEC-2 and REXX EXECs are supported as storage resident EXECs [RC = 4]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

EXECMAP

Use the EXECMAP command to list execs and System Product Editor macros in storage and in certain loaded saved segments. These segments are the CMS installation saved segment and segments built by the SEGGEN command that are loaded. See the *Application Development Guide for CMS* for details on the SEGGEN command.

Format

EXECMap EXMap	$[\textit{execname} \textit{ [exctype] }] \quad [(\textit{options...} [])]$ <p style="text-align: center;"> $[\textit{ * } \quad [\textit{ * }]]$ </p> <p>Options: [User] [SYSTEM] [SHARED]</p> $[\textit{SEGment} \left\{ \textit{segname} \right\}]$ <p style="text-align: center;"> $[\textit{ * }]$ </p> <p>NOSEGment</p> $[\textit{STACK} \left[\textit{FIFO} \right]]$ <p style="text-align: center;"> $[\textit{LIFO}]$ </p> <p>FIFO</p> <p>LIFO</p>
--------------------------------	---

Operands

execname

is the name of the storage-resident exec(s) to be listed. If an asterisk (*) is coded in this field, all exec names are used. The default is an asterisk.

exctype

is the type of the storage-resident exec(s) to be listed. If an asterisk (*) is coded in this field, all exctypes are used. The default is an asterisk.

If no operands are specified, the list contains all of the execs and editor macros in storage, the CMS installation saved segment, and loaded logical segments. Issuing EXECMAP with no operands is like issuing EXECMAP * *.

Options

User

indicates that the storage for the exec(s) was allocated from user free storage when the exec was loaded. Only the execs that satisfy the *execname* and *exctype* qualifications and that also have the USER attribute are listed. If neither USER, SYSTEM, nor SHARED is specified, then all execs that satisfy the *execname* and *exctype* qualifications are listed. Shared execs are not listed if SET INSTSEG is OFF.

System

indicates that the storage for the exec(s) was allocated from nucleus free storage when the exec was loaded. Only the execs that satisfy the *execname* and *execotype* qualifications and that also have the SYSTEM attribute are listed. If neither USER, SYSTEM, nor SHARED is specified, then all execs that satisfy the *execname* and *execotype* qualifications are listed. Shared execs are not listed if SET INSTSEG is OFF.

SHared

indicates that the exec(s) is affected by the SET INSTSEG command. Only the execs that satisfy the *execname* and/or *execotype* qualifications and that also have the SHARED attribute are listed. If neither USER, SYSTEM, nor SHARED is specified, then all execs that satisfy the *execname* and *execotype* qualifications are listed. Shared execs are not listed if SET INSTSEG is OFF.

SEGment

indicates that only execs in the specified segment are to be listed.

If neither SEGMENT or NOSEGMENT is specified, all execs will be listed, whether or not they are segment resident.

segname

the 1-8 character name of a logical segment. An asterisk (*) will list all execs that match the *execname* and *execotype* and reside in a logical segment.

NOSEGment

indicates that only execs that do not reside in a logical segment are to be listed.

If neither SEGMENT or NOSEGMENT is specified, all execs will be listed, whether or not they reside in a segment.

STACK [FIFO]**STACK LIFO**

specifies that the listed information should be placed in the program stack (for use by an exec or other program) instead of being displayed at the terminal. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked FIFO. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked LIFO. This option is equivalent to STACK LIFO.

Usage Notes

1. The exec information is listed alphabetically by exec name. For duplicate exec names, the first one listed is the one activated by commands specifying its name.
2. Use the EXECDROP command to delete a storage-resident exec or to discontinue use of an exec in a saved segment. To discontinue use of all shared execs temporarily, use the SET INSTSEG OFF command. Use the EXECLOAD command to load an exec into storage. Use the EXECSTAT command to determine the status of storage-resident execs.

Examples

To list all storage-resident execs with an exectype of XEDIT, then specify:

```
execmap * xedit
```

The following command:

```
execmap travel exec (fifo
```

stacks the map output FIFO for the storage-resident TRAVEL EXEC.

Responses

Name	Type	Usage	Records	Bytes	Attribute	Segname
execname	exectype	usage	records	bytes	USER	segname
.	SYSTEM	
.	SHARED	
.	

Messages and Return Codes

DMSEXM003E Invalid option: *option* [RC = 24]
 DMSEXM070E Invalid parameter *parameter* [RC = 24]
 DMSEXM109S Virtual storage capacity exceeded [RC = 104]
 DMSEXM415E Invalid character *char* in execid *execname exectype* [RC = 20]
 DMSEXM416W There are no *execname exectype*
 {SYSTEM|[or]USER|[or]SHARED} EXECs storage resident
 [RC = 28]
 DMSEXM1284T Non-recoverable error occurred in system data management
 routines. Re-IPL CMS.

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

EXECOS

Use the EXECOS command to reset the OS and VSAM environments under CMS without returning to the interactive environment. The EXECOS command can be invoked by specifying EXECOS without parameters or by *preceding* any CMS command with EXECOS.

Format

EXECOS	<i>[cmd [operand1 [operand2 ...operandn]]]</i>
---------------	--

Operands

[cmd [operand1 [operand2 ...operandn]]]

is a CMS command. If EXECOS precedes a CMS command, first the CMS command is processed, and then the EXECOS performs the OS reset function. The return code is that of the CMS command that was processed. A return code of -3 may indicate that the EXECOS command preceded an unknown CMS command. If EXECOS is specified without parameters, the OS environment is reset and the return code is zero.

Usage Notes

1. The EXECOS command is primarily intended for use in an EXEC 2 or a System Product Interpreter exec that either invokes several OS programs sequentially or invokes the same OS program repetitively.

Note: The cleanup of an OS or VSAM environment, if needed, is performed after each command has been processed from a CMS exec or after processing a command entered at your terminal. Therefore, in these two cases, the use of the EXECOS command is unnecessary.

2. The EXECOS command clears the following:

- STAE exits
- STIMER exits
- SPIE exits
- STAX exits
- TXTLIB directories
- MACLIB pointers
- SSTAT extensions
- LINKLISTS (LINKSTRT and LINKLAST)
- OS environment flags (OSSFLAGS)

If VSAM is running, VSAM cleanup is also done.

If the Vector Facility is in use, it is disabled to force a vector reset.

3. When processing in the VSAM environment, issue EXECOS prior to issuing CMS commands or user applications that run in the user area. Failure to do so could result in unpredictable errors and the overlay of VSAM control blocks.
4. When you request a reset of the OS environment after the execution of a CMS exec, the EXECOS command should *precede* the CMS exec command. For example:

```
&TRACE ALL
EXECOS EXEC VMFASM DMSSEB DMSSP
&EXIT
```

5. Whenever the CMS command, CP, must be used within an exec or EXEC 2 exec to transmit a CP command to the VM/SP control program environment, a request for a reset of the OS environment after the execution of the CP command requires that the EXECOS command precede the CMS CP command. For example:

```
&TRACE ALL  
EXECOS CP QUERY DASD  
&EXIT
```

EXECSTAT

Use the EXECSTAT command to verify the existence of an exec or System Product Editor macro in storage, on a disk, or in an SFS directory. The status is returned as a return code in register 15.

Format

EXECStat EXStat	$\left\{ \begin{array}{c} \textit{execname} \\ * \end{array} \right\} \left\{ \begin{array}{c} \textit{exectype} \\ * \end{array} \right\}$
----------------------------------	---

Operands

execname

is the name of the storage-resident exec(s) whose existence is to be verified. If an asterisk (*) is coded in this field, all storage resident execnames will be used.

exectype

is the type of the storage-resident exec(s) whose existence is to be verified. If an asterisk (*) is coded in this field, all exectypes are used.

Usage Notes

1. Specifying an asterisk (*) for both the execname and the exectype will verify the existence of any storage-resident execs. When you specify 'EXECSTAT * *', the return codes have the following meanings:

Code Meaning

0 There are storage-resident execs.

4 There are no storage-resident execs.

The contents of register 1 should be disregarded in this case.

2. To list the execs in storage and in a CMS installation saved segment, use the EXECMAP command. To remove an exec from storage or to discontinue use of an exec in a CMS installation saved segment, use the EXECDROP command. Use the SET INSTSEG OFF command to discontinue use of all SHARED execs temporarily. To load an exec into storage, use the EXECLOAD command.

Examples

To see if there are any storage-resident execs with the execname FILELIST, you would enter:

```
execstat filelist *
```

To see if there are any storage-resident execs with an exectype of XEDIT, you would enter:

```
execstat * XEDIT
```

Responses

The return codes and their meanings are:

Code Meaning

- 0 Exec will execute from storage and register 1 contains pointer to the fileblock.
- 4 Exec will execute from dasd and register 1 contains pointer to the FST.
- 8 Exec will not execute from storage and does not exist on dasd.

Messages and Return Codes

- DMSEXQ003E Invalid option: *option* [RC = 24]
- DMSEXQ042E No execid specified [RC = 24]
- DMSEXQ054E Incomplete execid specified [RC = 24]
- DMSEXQ070E Invalid parameter *parameter* [RC = 24]
- DMSEXQ109S Virtual storage capacity exceeded [RC = 104]
- DMSEXQ415E Invalid character *char* in execid *execname* *exectype* [RC = 20]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

EXECUPDT

Use the EXECUPDT command to apply updates to a System Product Interpreter source program and create an executable version of the program. This command can only be used with System Product Interpreter programs. See *VM/SP Application Development Guide for CMS* for information on how to set up a program for updating. See the *VM/SP System Product Editor Command and Macro Reference* for information about editing files in update mode with the System Product Editor.

EXECUPDT creates a variable-format, executable program from a fixed-format, sequenced source file. The file type of the source file must begin with a dollar sign (\$), for example \$EXEC or \$XEDIT. You do not enter this dollar sign when you enter the file type on the EXECUPDT command. EXECUPDT updates files in the same manner as the UPDATE command and accepts all UPDATE command options. In addition to the UPDATE command options, EXECUPDT has options that allow you to request removal of Support Identification (SID) codes from the file, include a log of applied updates with the file, or remove comments from the file to improve performance.

Format

EXECUPDT	$fn \left[\begin{array}{l} ft \\ \underline{\text{EXEC}} \end{array} \right] \left[\begin{array}{l} fm \\ * \end{array} \right] \left[(\text{options...}) \right]$ <p>Options:</p> $\left[\text{CTL } fn1 \right] \left[\begin{array}{l} \underline{\text{HISTory}} \\ \underline{\text{NOHISTory}} \end{array} \right] \left[\begin{array}{l} \underline{\text{COMPress}} \\ \underline{\text{NOCOMPress}} \end{array} \right] \left[\begin{array}{l} \underline{\text{COMMENTS}} \\ \underline{\text{NOCOMMENTS}} \end{array} \right]$ $\left[\underline{\text{ETMODE}} \right] \left[\begin{array}{l} \underline{\text{SID}} \\ \underline{\text{NOSID}} \end{array} \right] \left[\underline{\text{NOUPdate}} \right]$
-----------------	---

Operands

fn ft fm

is the file identifier of the source input file. The file must consist of fixed-format records with a record length between 80 and 255, inclusive, with sequence numbers in the last eight columns. If the file type or file mode is omitted, EXEC and * are assumed, respectively.

Options

CTL

specifies that a file named "fn1" with a file type of CNTRL is an update control file that controls the application of multiple update files to the source input file.

HISTory

specifies that a file named 'fn' with a file type of UPDATES is to be appended to the updated source file as a System Product Interpreter comment. This file, 'fn UPDATES', is created by the UPDATE command.

NOHISTory

specifies that the UPDATES file created by the UPDATE command should not be appended to the updated source file. This is the default.

COMPRESS

specifies that comment lines in the source file that start in column 1 and begin with /*! should be removed from the updated source file to improve performance. This is the default.

Note: You identify those comments that are extraneous (for instance, a prolog) by putting an exclamation point (!) after the /* that begins the comment. Remember that you must leave a comment as the first statement so VM/SP can identify it as a System Product Interpreter program.

NOCOMPRESS

specifies that comments are not to be removed from the updated source file.

COMMENTS

specifies that comments are not to be removed from the source file except those removed by the COMPRESS option. This is the default.

NOCOMMENTS

specifies that all comments and leading blanks are to be removed from the source file. One comment line containing the execname and exectype is inserted at the beginning of the file. Do not specify NOCOMMENTS with the COMPRESS option.

Note: If you use NOCOMMENTS on an EXEC that uses data stored within comments, the updated file may not execute properly.

ETMODE

specifies that the source file contains DBCS characters and that shift-in and shift-out characters should be paired while comments are being removed. ETMODE only functions when specified with NOCOMMENTS.

SID

specifies that the last seventeen columns of the input file contain sequence number and SID code fields which are to be removed from the updated source file. The sequence number field is the last eight columns of the input file, the SID code field is the nine columns before the sequence number.

NOSID

specifies that the input file does not contain a SID code. Only the sequence numbers in the last eight columns of the input file are to be removed from the updated source file. This is the default.

NOUPDATE

specifies that no update files are to be applied before building the updated source file. Specify this option when converting a source file which does not yet have updates. You can also use the CTL option to convert a file with no updates. The EXECUPDT command ignores the HISTORY option if NOUPDATE is specified.

Usage Notes

1. You may also use any other options accepted by the UPDATE command. See the description of the CMS UPDATE command for a complete list of available options.
2. If you want to issue EXECUPDT from an exec program, you should precede it with the EXEC command; that is, specify
exec execupdt

Example

If you want to create an executable version of the source file MYFILE \$XEDIT, and remove comments from the file, you would enter:

```
execupdt myfile xedit a (comp noup
```

The option COMP, for COMPRESS, is actually the default; therefore, you do not have to specify it. The option NOUP, for NOUPDATE, means that update file processing is ignored and that a file named myfile xedit is produced from myfile \$xedit without the comments.

Messages and Return Codes

DMSWUP002E File *fn ft fm* not found [RC = 28]
DMSWUP054E Incomplete fileid specified [RC = 24]
DMSWUP419E *fn ft* has an error with quote/comment nesting. A quote is|A comment is|*n* comments are open at the end of the program. [RC = 8]
DMSWUP637E Missing value for the CTL option [RC = 24]
DMSWUP649E Extraneous parameter *parameter* [RC = 24]
DMSWUP671E Error updating *fn ft fm*; rc = *nn* from XEDIT [RC = 100]
DMSUPD001E No filename specified [RC = 24]
DMSUPD002E File [*fn [ft [fm]]*] not found [RC = 28]
DMSUPD003E Invalid option: *option* [RC = 24]
DMSUPD007E File *fn ft fm* is not fixed, 80-character records [RC = 32]
DMSUPD010W Premature EOF on file *fn ft fm*--sequence number *seqno* not found [RC = 12]
DMSUPD024E File *fn ft fm* already exists [RC = 28]
DMSUPD048E Invalid mode *mode* [RC = 24]
DMSUPD065E *option option* specified twice [RC = 24]
DMSUPD066E *option1* and *option2* are conflicting options [RC = 24]
DMSUPD069E Filemode *mode* not accessed [RC = 36]
DMSUPD070E Invalid parameter *parameter* [RC = 24]
DMSUPD104S Error *nn* reading file *fn ft fm* from disk or directory [RC = 100]
DMSUPD105S Error *nn* writing file *fn ft fm* on disk or directory [RC = 100]
DMSUPD174W Sequence error introduced in output file: *seqno1* to *seqno2* *seqno1* to *seqno2* [RC = 8]
DMSUPD176W Sequencing overflow following sequence number *seqno* [RC = 8]
DMSUPD179E Missing or duplicate MACS card in control file *fn ft fm* [RC = 32]
DMSUPD181E No update files were found [RC = 40]
DMSUPD182W Sequence increment is zero [RC = 8]
DMSUPD183E Invalid {CONTROL|AUX} file control card [RC = 32]
DMSUPD184W ./S not first card in update file--ignored [RC = 12]
DMSUPD185W Invalid character in sequence field *seqno* [RC = 12]
DMSUPD186W Sequence number *seqno* not found [RC = 12]
DMSUPD187E Option STK invalid without CTL [RC = 24]
DMSUPD207W Invalid update file control card [RC = 12]
DMSUPD210W Input file sequence error: *seqno1* to *seqno2* [RC = 4]
DMSUPD299E Insufficient storage to complete update [RC = 41]
DMSUPD300E Insufficient storage to begin update [RC = 41]
DMSUPD361E Filemode *mode* is not a CMS disk or directory [RC = 36]
DMSUPD653E Error executing *command* rc = *nn* [RC = 40]

FETCH

Use the FETCH command in CMS/DOS to load an executable phase into storage for execution.

Format

FETCh	<i>phasename</i> [(options...[])] <u>Options:</u> [START] [COMP] [ORIGIN <i>hexloc</i>]
--------------	--

Operands

phasename

is the name of the phase to be loaded into virtual storage. CMS searches for the phase:

- In a VSE private core image library, if IJSYSCL has been defined.
- In CMS DOSLIBs that have been identified with the GLOBAL command.
- In the VSE system core image library, if you specified the mode letter of the VSE system residence on the SET DOS ON command line.

Options

START

specifies that once the phase is loaded into storage, execution should begin immediately.

COMP

specifies that when the phase is to be executed, register 1 should contain the address of its entry point. (See Usage Note 5.)

ORIGIN *hexloc*

fetches the program and loads it at the location specified by *hexloc*; this location must be in the CMS user area below the start of the CMS nucleus. The location, *hexloc*, is a hexadecimal number of up to eight characters. (See Usage Note 6.)

Usage Notes

1. If you do not use the START option, FETCH displays a message at your terminal indicating the name of the phase and the storage location of its entry point. At this time, you can set address instruction stops for testing. To continue, issue the START command to initiate execution of the phase just loaded.
2. The fetch routine is also invoked by supervisor call (SVC) instructions 1, 2, 4, or 65. The search order for executable phases is the same as listed above. An exception to this search order is CMS DOSLIBs, as they are not searched.
3. If you want to fetch a phase from a private core image library, you must issue an ASSGN command for the logical unit SYSCLB and define the library in a DLBL command using the ddname IJSYSCL. For example:

FETCH

```
assgn sysclb c
dlbl ijsyscl c dsn core image lib (sysclb perm
```

4. Phase fetched from VSE core image libraries must have been link-edited with ACTION REL.
5. CMS uses the COMP option when it fetches the DOS PL/I compiler because that compiler expects register 1 to contain its entry point address. This option is not required when you issue the FETCH command to load your own programs.

When CMS starts executing a phase that has COMP specified, the

```
DMSLI0740I      Execution begins ...
```

message is not displayed.

6. The ORIGIN option is used by the VSAMGEN installation exec procedure to load nonsharable modules on a segment boundary. It is not required when you issue the FETCH command to load your own programs, unless you want to load them at a location other than 20000.
7. The FETCH command should only be used with the START command to execute a VSE program. It should not be used with GENMOD to attempt to create an executable CMS module file.
8. Multiphase program support is different in CMS/DOS than in VSE. The core image directory is not searched for multiphase programs. Thus the value of HIPROG in BGC0M reflects only the ending address of the longest phase loaded, not that of the phase in the library that has the highest ending address.

Example

To load the executable phase MYFILE into your CMS virtual machine and immediately begin execution of the program, you would enter:

```
fetch myfile (start
```

Responses

```
DMSFET710I      Phase phase entry point at location hexloc
```

This message is issued when the START option is not specified. It indicates the virtual storage address at which the phase was loaded.

```
DMSLI0740I      Execution begins ...
```

This message is issued when the START option is specified; it indicates that program execution has begun.

Messages and Return Codes

DMSFCH016E	No private core image library found [RC=28]
DMSFCH104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC=100]
DMSFCH109S	Virtual storage capacity exceeded [RC=104]
DMSFCH113S	Disk(<i>vdev</i>) not attached [RC=100]
DMSFCH115E	Phase load point less than <i>vstor</i> [RC=40]
DMSFCH411S	Input error code <i>nn</i> on SYS <i>aaa</i> [RC= <i>rc</i>]
DMSFCH623S	Phase cannot be loaded at location <i>hexloc</i> --this area is available for system use only [RC=88]
DMSFCH777S	DOS partition too small to accommodate fetch request [RC=104]
DMSFET003E	Invalid option: <i>option</i> [RC=24]
DMSFET004E	Phase <i>phase</i> not found [RC=28]
DMSFET029E	Invalid parameter <i>parameter</i> in the option <i>option</i> field [RC=24]
DMSFET070E	Invalid parameter <i>parameter</i> [RC=24]

DMSFET098E No phase name specified [RC = 24]
DMSFET099E CMS/DOS environment not active [RC = 40]
DMSFET623S Phase cannot be loaded at location *hexloc*--this area is available
for system use only [RC = 88]

FILEDEF

Use the FILEDEF command to establish data definitions for OS *ddnames*, to define files to be copied with the MOVEFILE command, or to override default file definitions made by the assembler and the OS language processors.

Format

Filedef	<pre> { ddname * } Terminal [(optionA optionB optionE [])] PRinter [(optionA optionB OPTCD J [])] PUrch [(optionA optionB [])] Reader [(optionA optionB [])] DISK [fn ft [fm]] [(optionA optionB optionC [])] [FILE ddname [A1]] DISK [[[fn ft [fm]] { DSN ? DSN qual1 qual2 ... DSN qual1.qual2 ... } [(optionA optionB optionC [])] DISK vaddr DUMMY [(optionA optionB [])] TAP n [LABOFF BLP [n] SL [n][VOLID valid] [(DISP MOD optionF[])] SUL [n][VOLID valid] NL [n] NSL filename] [(optionA optionB optionD [])] GRAF vdev [(optionA [])] CLEAR </pre>
----------------	--

Filedef	<p>OptionA: [PERM] [CHANGE NOCHANGE]</p> <p>OptionB: [RECFM <i>a</i>] [LRECL <i>nnnnn</i>] [BLOCK <i>nnnnn</i> BLKSIZE <i>nnnnn</i>]</p> <p>OptionC: [KEYLEN <i>nnn</i>] [XTENT <i>nnnnn</i>] [LIMCT <i>nnn</i>] [OPTCD <i>a</i>] [DISP MOD] 50 [MEMBER <i>membername</i>] [CONCAT] [DSORG {PS PO DA}]</p> <p>OptionD: [7TRACK 9TRACK 18TRACK] [TRTCH <i>a</i>] [DEN <i>den</i>] [LEAVE] [NOEOV] [ALT {TAP <i>n</i> } {<i>vdev</i> }]</p> <p>OptionE: [UPCASE LOWCASE]</p> <p>OptionF: [SYSPARM {(<i>string</i>) } {(?)}]</p>
----------------	---

Operands

ddname

nn

*

is the name by which the file is referred to in your program. The *ddname* may be from one to eight alphanumeric characters, but the first character must be alphabetic or national. If a number *nn* is specified, it is translated to a FORTRAN data definition name of FT*nn*F001. An asterisk (*) may be specified with the CLEAR operand to indicate that all file definitions not entered with the PERM option should be cleared.

Devices:

Terminal

is your terminal (terminal I/O must not be blocked). Terminal input will be truncated to the console input buffer length of 130 characters.

PRinter

is the spooled printer.

PUnch

is the spooled punch.

Reader

is the spooled card reader (card reader I/O must not be blocked).

DISK

specifies that the virtual I/O device is a disk or SFS directory. As shown in the format, you can choose one of two forms for specifying the DISK operand. Both forms are described in "Using the FILEDEF DISK Operand."

DUMMY

indicates that no real I/O takes place for a data set.

TAP [n]

is a magnetic tape. The following symbolic names for a tape drive are supported and represent these virtual control units:

Symbolic Name	Virtual Address	Symbolic Name	Virtual Address
TAP0	180	TAP8	288
TAP1	181	TAP9	289
TAP2	182	TAPA	28A
TAP3	183	TAPB	28B
TAP4	184	TAPC	28C
TAP5	185	TAPD	28D
TAP6	186	TAPE	28E
TAP7	187	TAPF	28F

If *n* is not specified, FILEDEF uses the existing TAP n device for the specified *ddname*. TAP defaults to TAP2 if there is no existing definition for the specified *ddname*, or if the existing device was not TAP n . TAP0 through TAP7 are at one virtual control unit and TAP8 through TAPF are at another virtual control unit. You can also specify the type of label processing you want on your tape. Specifying label processing is discussed in "Using the FILEDEF TAP n operand."

GRAF

specifies that the virtual I/O device is a Graphic Display.

vdev

is the virtual device address of the attached graphic display. Valid addresses are:

- 0001 through FFFF for a 370/XA mode virtual machine
- 001 through 5FF for a VM/SP virtual machine in basic control mode
- 001 through FFF for a System/370 mode virtual machine and a VM/SP virtual machine in extended control mode.

On VM/SP and System/370 mode virtual machines you can supply a leading zero. In the preceding description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. Valid addresses in both environments are described to help you plan and develop applications that will run in both environments.

CLEAR

removes any existing definition for the specified *ddname*. Clearing a *ddname* before defining it ensures that a file definition does not exist and that any options previously defined with the *ddname* no longer have effect.

Options

Whenever an invalid option is specified for a particular device type, an error message is issued. Table 7 on page 190 shows valid options for each device type.

Table 7. Valid File Characteristics for Each Device Type of the FILEDEF Command					
Options	OPERANDS READER, PUNCH, PRINTER	TERMINAL	TAP ⁿ	DISK DUMMY ¹	GRAF
ALT			X ⁴		
BLOCK, BLKSIZE	X	X	X ⁶	X	
CHANGE, NOCHANGE	X	X	X	X	X
CONCAT				X	
DEN			X		
DISP MOD			X ⁴	X	
DSORG				X	
KEYLEN				X ²	
LEAVE			X		
LIMCT				X ²	
LOWCASE, UPCASE		X			
LRECL	X	X	X ⁶	X	
MEMBER				X	
NOEOV			X		
OPTCD	X ⁵			X ²	
PERM	X	X	X	X	X
SYSARM			X ⁴		
RECFM	X	X	X	X	
TRTCH			X ³		
XTENT				X ²	
7TRACK, 9TRACK, 18TRACK			X		

-
- 1 No options may be necessary but all disk options are accepted.
 - 2 This option is meaningful only for BDAM files.
 - 3 This option is for 7-track tapes only.
 - 4 This option is for SL tapes only.
 - 5 This option is for Printer only.
 - 6 This should be used for OS compatibility of output.

PERM

retains the current definition until it either is explicitly cleared or is changed with a new FILEDEF command with the CHANGE option. If PERM is not specified, the definition is cleared when a FILEDEF * CLEAR command is executed.

CHANGE

merges the file definitions whenever a file definition already exists for a *ddname* and a new FILEDEF command specifying the same *ddname* is issued; the options associated with the two definitions are merged. Options from the original definition remain in effect unless duplicated in the new definition. New options are added to the option list.

NOCHANGE

retains the current file definition, if one exists, for the specified *ddname*. With this option, the system stops further processing (error checking, scanning, etc.) of the new FILEDEF command if a file definition exists for the specified *ddname*.

RECFM a

is the record format of the file, where “a” can be one of the following:

a	Meaning
F	Fixed length
FB⁷	Fixed blocked
V	Variable length
VB⁷	Variable blocked
U	Undefined
FS,FBS	Fixed length, standard blocks
VS,VBS	Variable length, spanned records

The values below, **A** and **M**, are carriage control characters. They may be combined with any of the above values for “a.” For example, specifying **FBA** gives fixed block records with ASA control characters. Specifying **VBM** gives variable blocked records with machine codes.

A⁸	ASA control characters
M⁹	Machine codes

LRECL nnnnn

is the logical record length (nnnnn) of the file, in bytes. LRECL should not exceed 32760 bytes for fixed length records or 32756 for variable length records because of OS restrictions.

BLOCK nnnn**BLKSIZE nnnnn**

is the logical block size (nnnnn) of the file, in bytes. BLOCK should not exceed 32760 bytes because of OS restrictions. If both BLOCK and BLKSIZE options are specified, the value of nnnnn for BLOCK is used and BLKSIZE is ignored.

⁷ **FB** and **VB** should not be used with TERMINAL or READER devices.

⁸ **A** may be used with TERMINAL devices, but simulation is limited to the characters “blank” (single space), “0,” (double space), “-” (triple space), and “1” (page eject, simulated as double space). All other control characters are treated as a single space.

⁹ **M** should not be used with TERMINAL devices.

KEYLEN *nnn*

is the size (*nnn*) of the key (in bytes). The maximum value accepted is 256.

XTENT *nnnnn*

is the number of records (*nnnnn*) in the extent for the file. The default is 50. The maximum value is 16,777,215.

LIMCT *nnn*

is the maximum number of extra tracks or blocks (*nnn*) to be searched. The maximum value is 256.

OPTCD *a*

is the direct access search processing desired. The variable "a" may be any combination of up to three of the following: (A and R are mutually exclusive.)

Code	DASD Search
A	Actual device addressing
E	Extended search
F	Feedback addressing
R	Relative block addressing

OPTCD *J*

is valid only for the Printer. When the virtual printer is a 3800, 'J' indicates to QSAM and BSAM that the output line contains a TRC (Table Reference Character) byte.

Note: The KEYLEN, XTENT, LIMCT, and OPTCD options should only be used with BDAM, QSAM, or BSAM files.

DISP MOD

positions the read/write pointer after the last record in the file. This option should only be used for adding records to the end of a file and only for standard label tapes. When you add records to the end of a file, the file must be on a disk or directory accessed as read/write. If a disk or directory is an extension of another disk or directory, the extension is automatically read/only and you cannot write to it. During multi-volume tape processing this is valid only with the currently mounted volume; no volume switching is done.

MEMBER *membername*

allows you to specify the name of a member of an OS partitioned data set, or a CMS simulated partitioned data set; member name is the name of the PDS member. With CMS simulated partitioned data sets, OS macros can only be used to read members.

CONCAT

allows you to assign the same *ddname* to two or more OS libraries so that you can refer to them in a single GLOBAL command. You may concatenate libraries with file types of MACLIB and LOADLIB.

You cannot issue multiple FILEDEF commands with the CONCAT option for LOADLIBs and MACLIBs that have the same file names but are on different disks or directories. Only the information for the last FILEDEF is retained.

Any file format options you specify in the first FILEDEF command line remain in effect for subsequently concatenated libraries. For a detailed description of concatenated macro libraries, see *VM/SP Application Development Guide for CMS*.

DSORG PS
DSORG PO
DSORG DA

is the data set organization: physical sequential (PS), partitioned (PO), or direct access (DA).

7TRACK
9TRACK
18TRACK

is the tape setting. The tape device mode is not checked or set by FILEDEF. Use the TAPE command MODESET option to set the mode of a tape.

TRTCH a

is the tape recording technique for 7-track tapes. Use the following chart to determine the value of "a" for 7-track tapes.

a	Parity	Converter	Translator
O	odd	off	off
OC	odd	on	off
OT	odd	off	on
E	even	off	off
ET	even	off	on

The default value of TRTCH is OC.

DEN den

is tape density: den can be 200, 556, 800, 1600, 6250, or 38K BPI (bytes per inch). If 200 or 556 are specified, 7TRACK is assumed. If 800, 1600, or 6250 are specified 9TRACK is assumed. If 38K is specified, 18TRACK is assumed. If a TAPE command was issued with the MODESET option and MODESET was not specified on any subsequent command, the tape drive will contain the density (mode) specified by TAPE MODESET. If no MODESET was specified, the drive will default to the highest density available; for example, DEN 6250 for a 1600/6250 dual density 3420 drive. For additional information, see the MODESET option of the TAPE command.

The following densities are allowed for the given track sizes.

7track

200, 556, 800

9track

800, 1600, 6250

18track

38K

UPCASE

translates all terminal input data to uppercase.

LOWCASE

retains all terminal input data as typed in.

LEAVE

is only valid for TAPn files that are SUL or SL (standard label). With this option selected, the tape is not moved before label processing. If LEAVE is not specified, tapes with files specified as SL or SUL are rewound and then positioned before the files are processed.

NOEOV

is only valid for TAPn files. With NOEOV selected, there is no automatic limited end-of-volume processing when end of tape is sensed on output. Under OS simulation for standard labelled tapes, if NOEOV is specified, it is ignored during end-of-volume processing. See the section “CMS Tape Label Processing” in the *VM/SP CMS User's Guide* for a description of end-of-volume processing.

ALT TAPn

ALT vdev

is only valid for TAPn files that are SL (standard label). This option specifies an alternate tape drive to be used when an EOVS condition occurs on the primary tape drive. It allows the next volume of a multi-volume tape file to be mounted on an alternate drive at the same time as the first volume. Processing switches between drives at each EOVS condition and continues until an EOT condition is encountered.

The TAPn specifies the symbolic tape identification. The vdev specifies the virtual device address of the tape. The following symbolic names and virtual device addresses are supported:

Symbolic Name	Virtual Address	Symbolic Name	Virtual Address
TAP0	180	TAP8	288
TAP1	181	TAP9	289
TAP2	182	TAPA	28A
TAP3	183	TAPB	28B
TAP4	184	TAPC	28C
TAP5	185	TAPD	28D
TAP6	186	TAPE	28E
TAP7	187	TAPF	28F

Both the primary and alternate tape drives must have compatible characteristics; that is, they must have an equal track setting and must record at the same density.

Usage Notes

1. If you do not issue a FILEDEF command for an OS input or output file, CMS uses the *ddname* on the DCB macro to issue the following default file definition:

```
FILEDEF ddname DISK FILE ddname A1
```

See “Usage Notes” under the discussion of the ASSEMBLE command for information on the default file definitions made by the assembler.
2. To identify VSE files for VSE program execution or to identify VSAM data sets for either OS or VSE program execution, you must use the DLBL command.
3. A file definition established with the FILEDEF command remains in effect until explicitly changed or cleared. The system clears file definitions under the following circumstances:
 - When the assembler or any of the language processors are invoked. (Note that FILEDEF definitions entered with the PERM option are not cleared.)
 - When a program abends or when you issue the Immediate command HX to halt command or program execution.

4. Do not attempt to clear a FILEDEF in a program after the corresponding DCB has been opened - you must do this before the DCB has been closed. The OS macros rely on information set up by FILEDEF. Therefore, if this information is cleared, the macros are not able to function properly, and unpredictable results can occur.
5. The FILEDEF command does not supply default values for LRECL and BLKSIZE. As under OS, if DCB information is unavailable when a file is opened, an open error is issued for the file. The following chart summarizes the results at OPEN time of specifying LRECL and BLKSIZE options.

BLKSIZE	LRECL	Results
Not Specified	Not Specified	If the input file exists, the item length (or item length + 4 for variable length records) becomes the BLKSIZE.
Specified	Not Specified	LRECL = BLKSIZE (or LRECL = BLKSIZE-4, for variable-length records).
Not Specified	Specified	BLKSIZE = LRECL (or BLKSIZE = LRECL + 4, for variable-length records).
Specified	Specified	The values specified are used.

If V or VB is specified for RECFM, LRECL must be at least 4 bytes less than BLKSIZE and LRECL must be at least 4 bytes greater than the largest record of the file. If VS or VBS is specified for RECFM, LRECL can exceed the specified BLKSIZE, but LRECL should not exceed a maximum value of 32756 because of OS restrictions.

VSE sequential (SAM) files do not contain BLKSIZE, LRECL, or RECFM specifications. These options must be specified by a FILEDEF command or DCB statement if OS macros are used to access VSE files. Otherwise the defaults, BLKSIZE = 32760 and RECFM = U, are assumed. LRECL is not used for RECFM = U files.

6. When copying a variable length data set (RECFM = V or VB) from an OS disk to a CMS disk or SFS directory, the logical record length (LRECL) of the file that is created on the CMS disk or SFS directory is equal to the size of the largest record in the data set being copied. If the file that is being created has a file mode of 4, the logical record length will be equal to the LRECL of the largest record plus 8 bytes. The actual LRECL of the new file can be determined by using the CMS LISTFILE command.
7. There is an auxiliary processing option for FILEDEF that is only valid when FILEDEF is executed by an internal program call: this option cannot be entered as a terminal command. The option, AUXPROC addr, allows an auxiliary processing routine to receive control during I/O operations. For details on how to use this option of the FILEDEF command, see the manual, *VM/SP Application Development Guide for CMS*.
8. If a FILEDEF command is issued with a DDNAME that matches a current DDNAME defined by a previous FILEDEF command and the devices are the same, the file name, file type, file mode, and options previously specified remain in effect, unless re-specified by the new FILEDEF command. If the devices are not the same, all previous specifications are removed.

9. CMS supports one virtual reader at address 00C, one virtual punch at address 00D, and one virtual printer at address 00E. When you issue a CMS command or execute a program that uses one of these unit record devices, the device must be attached at the virtual address indicated.

If a program has two or more data control blocks (DCBs) with different *ddnames* open for the same unit record devices, records from different files may be mixed together into one file. Separate files are not maintained.

10. To copy data between shared (e.g. 3420) and non-shared (e.g. 3480) tape subsystems, assign each tape device to a different virtual control unit. For example:

```
FILEDEF IN TAP0 (18TRACK      *TAP0 is assigned to 180
FILEDEF OUT TAP8 (9TRACK     *TAP8 is assigned to 288
MOVEFILE IN OUT
```

or...

```
FILEDEF IN TAPA (18TRACK     *TAPA is assigned to 28A
FILEDEF OUT TAP1 (9TRACK     *TAP1 is assigned to 181
MOVEFILE IN OUT
```

11. If the FILEDEF command is entered with no operands, a list of current definitions is displayed.
12. FILEDEF uses the extended plist for processing the DSN *qual1* [*qual2...qualn*] parameter. If you are calling FILEDEF from an assembler language program and using DSN *qual1*[*qual2...qualn*], you should supply an extended plist. *VM/SP Application Development Guide for CMS* has more information on how an assembler language program can supply an extended plist.
13. If you are calling FILEDEF from an assembler language program and using the SYSPARM option, you must supply an extended plist. Use the CMSCALL macro to do this; refer to the description of the CMSCALL macro in the *VM/SP Application Development Reference for CMS* for more details.

Using the FILEDEF DISK Operand There are three general forms for specifying the DISK operand in a FILEDEF command.

1. If you specify the first form:

```
FILEDEF ddname DISK fn ft [fm]
```

fn and *ft* (file name and file type) are assumed to be a CMS file ID. If *fm* is the file mode of an OS disk, *fn* and *ft* are assumed to be the only two qualifiers of an OS data set name. If *fm* is specified as an asterisk, (*) then all accessed disks and directories are searched.

You cannot use this form unless the OS data set name or VSE file ID conforms to the OS naming convention (1- to 8-byte qualifiers separated by periods, to a maximum of 44 characters, including periods). Also, the data set name can have only two qualifiers; otherwise, you must use the DSN ? or DSN *qual1...qualn* form. For example, if the OS data set name or VSE file ID is TEST.SAMPLE.MAY, you enter:

```
FILEDEF MINE B1 DSN TEST SAMPLE MAY
```

-- or --

```
FILEDEF MINE B1 DSN TEST.SAMPLE.MAY
```

-- or --

```
FILEDEF MINE B1 DSN ?
TEST.SAMPLE.MAY
```

If the OS data set name or VSE file ID is TEST.SAMPLE, then you may enter:

```
FILEDEF MINE DISK TEST SAMPLE B1
```

2. The second form of the DISK operand is used only with OS data sets and VSE files:

```
FILEDEF ddname DISK fn  ft      fm DSN ?
                        FILE ddname A1 DSN qual1 [qual2...]
                        DSN qual1 [.qual2...]
```

This form allows you to enter OS and VSE file identifications that do not conform to OS data set naming conventions. The DSN operand corresponds to the DSN parameter on the OS DD (data definition) statement. There are many ways you can specify this form:

- FILEDEF *ddname* DISK *fn ft fm* DSN *qual1* [*qual2...*]

-- or --

- FILEDEF *ddname* DISK *fn ft fm* DSN *qual1* [*.qual2...*]

The above forms of the FILEDEF command associate the CMS file name and file type you specify with the OS data set name or VSE file ID specified following the DSN operand. Once it is defined, you can refer to the OS data set name or VSE file ID by using the CMS file name and file type. If you omit DISK, *fn*, *ft*, and *fm*, the default values are FILE *ddname* A1.

- FILEDEF *ddname* DSN ?

This form of the FILEDEF command allows you to specify the OS data set name or VSE file ID interactively. Using this form, you can enter an OS data set name or VSE file ID containing embedded special characters such as blanks. If you use this form, the default file name and file type for your file, FILE *ddname*, is the CMS file name and file type associated with the OS data set name or VSE file ID. The file mode for this form is always the default, A1.

To use the interactive DSN operand, you key in DSN ?; CMS then requests that you enter the OS data set name or DOS file ID exactly as it appears in the data set or file. Do not omit the periods that separate the qualifiers of an OS data set name, but do not insert periods where they do not appear.

```
qual1 [.qual2...]
```

where *qual1.qual2...* are the qualifiers of the OS data set name or VSE file ID. When you use this form, you must code the periods separating the qualifiers.

- FILEDEF *ddname* mode DSN *qual1* [*qual2...*]

-- or --

- FILEDEF *ddname* mode DSN *qual1* [*.qual2...*]

This form allows you to specify the OS data set name or VSE file ID explicitly. The default value for the file name and file type is FILE *ddname*. When you use this form, you can use periods to separate the qualifiers. If the command is entered with a blank separating the qualifiers, FILEDEF replaces them with periods. For example, for an OS data set or VSE file named MY.FILE.IN, you enter:

FILEDEF *ddname* B1 DSN MY FILE IN

-- or --

FILEDEF *ddname* B1 DSN MY.FILE.IN

3. FILEDEF *ddname* DISK *vaddr*

This form associates a *ddname* with a virtual minidisk address. The *ddname* is the name of the virtual minidisk referred to in your program. It is intended to be used for a minidisk for which a RESERVE command has been issued. This form cannot be used as a regular FILEDEF for OS simulation. An open issued for such a *ddname* would fail. The *vaddr* is the virtual address of the device referred to in your program by *ddname*.

Using the FILEDEF TAPn Operand: When you define a tape file with the FILEDEF command, you can specify the type of label processing to be done for the file. You do this by specifying a second operand after the word TAPn. The operands that you may specify and their meanings are:

- LABOFF** indicates that there is no CMS tape label processing for this tape file. LABOFF is the default. The tape is not positioned if this operand is specified.
- BLP** indicates that the system is to bypass label processing but that the tape is to be positioned before the file is processed.
- SL** indicates that you are using IBM standard labels.
- SUL** indicates that you are using standard user labels (not processed for MOVEFILE).
- NL** indicates that your tape has no IBM standard labels. (Do not use this operand if your tape has a VOL1 label. A file on it will not be opened.)
- NSL** indicates that you are using nonstandard labels.

For the operands BLP, SL, and SUL:

- n* indicates the position of the file on a multfile volume. When *n* is not specified, the default is 1.

For SL and SUL files:

- valid* specifies a 1- to 6-character volume serial number to be verified by reading the VOL1 label on the tape. If not specified in FILEDEF, volume IDs may be specified on a LABELDEF command. If specified on both commands, the more recent specification is used. VALID is only valid for SL or SUL tape files. If VALID is not specified, the volume label on the tape is not checked.

For SL files:

- DISP MOD** adds records to tape files for standard label tapes only.

```
filedef file a tap1 sl (disp mod
```

When the file is opened (output), the tape is positioned at the end of the file, ready to add new records. During multi-volume tape processing this is valid only with the currently mounted volume; no volume switching is done.

When you use DISP MOD, you must specify the FID option with the LABELDEF command. The FID you specify must match the file ID of the tape file to which you are adding records. This helps ensure that

the records are added to the correct tape file. If the FID does not match the tape file ID, message DMSTLM434E is issued.

SYSPARM passes the address of the character value *string* to DMSTVI (an interface routine). The maximum length of the string is determined by the total length of command. The command cannot exceed 130 characters unless the command is issued from a program and an extended plist is used. In this case, the string can be up to 65,535 bytes. If longer, it is truncated. The string cannot contain blanks or parentheses.

If you want to enter a string with blanks or parentheses, use the SYSPARM (?) format. When you enter this format, you are prompted with

ENTER SYSPARM:

You can enter up to 130 characters. If longer, the string is truncated. You can omit the string's right parenthesis if SYSPARM is the last option specified.

For the NSL operand:

filename is required for NSL files. It is the file name of a file that contains a routine for processing nonstandard labels. The file name must be that of a TEXT or MODULE file. If you have both a MODULE and TEXT file with this name, the MODULE file is used. MODULE files must be created so that they start at an address that does not allow them to overlay a user program if they are to be used for NSL routines. See the section "Tape Labels in CMS" in the *VM/SP CMS User's Guide* for details on writing routines to process nonstandard labels.

Unless LRECL is specified on the output FILEDEF, the EOVS2/EOF2 label contains the record length of the last record written for record format V or VB tape files. Files with a record format F or FB contain zero unless LRECL is specified. LRECL is important for OS compatibility. You can define a file on tap2 with standard labels by using the following command:

```
filedef filea tap2 sl volid dept10
```

When this tape file is opened, CMS checks to see that it has a VOL1 label with a volume serial number of dept10.

To specify the second file on the same tape, use

```
filedef filea tap2 sl 2 volid dept10
```

The same file could be defined as having no labels by using

```
filedef filea tap2 blp 2
filedef filea tap2 nl 2
```

If you use the above specification, your tape must not contain IBM standard labels. NL causes CMS to read your tape when you try to open a file on it and checks to see if the tape contains a VOL1 label as its first record. If a VOL1 label is there, CMS does not open your tape file.

If you specify

```
filedef filea tap2 blp 2
```

FILEDEF

CMS positions the tape to the second file, but does not check to see if the tape has a label.

Note: If you mount a blank tape and specify NL, the tape will run off the end of the reel. Write a tape mark to prevent this from occurring.

To define a tape file with nonstandard labels, use the following command:

```
filedef filea tap2 nsl nonstd
```

The routine NONSTD must exist as a TEXT or MODULE file and be able to process the particular nonstandard labels you are using for your tapes.

If you defined filea with no label parameter at all, for example,

```
filedef filea tap2
```

there is no label processing or positioning before the data in filea is processed.

When you use the options DEN, TRTCH, 7TRACK, 9TRACK, or 18TRACK to set the mode of an output file, the tape must be at load point. This is due to a hardware restriction which allows the mode of a tape drive to be reset only when the tape is at load point. If the tape is not at load point when the data is written, the tape will be written at the current mode of the tape drive, and not the mode specified in the FILEDEF command. It is important to note that OPEN macro processing may position the tape beyond the load point (depending on the type of label processing is requested) before the data is written, and therefore the mode would not be reset. See the "CMS TAPE Command" Usage Notes for more information.

Read the section "Tape Labels in CMS" in the *VM/SP CMS User's Guide* before you write programs that handle labeled tapes.

Use the LEAVE and NOEOV options for tape files only.

LEAVE indicates that a tape containing standard-label files is not to be moved before label processing. Using this option prevents CMS from rewinding the tape and checking the VOL1 label as it otherwise does for SL and SUL files. The command:

```
filedef fileb tap1 sl (leave
```

defines a tape file on tap1 but tells CMS not to position the tape before processing the labels for fileb. Note that you must position the tape properly yourself before using the LEAVE option. LEAVE may be used with SL, SUL, and BLP. However, it has no effect if used with NL. NL tapes are always rewound and positioned before a file on them is opened (even if you specify LEAVE).

Use the LEAVE option with multifile volumes where rewinding and repositioning a tape before processing each file is inefficient. You must not move the tape between files if you use this option. Note that for BLP files you can obtain the effect of LEAVE by defining the file as LABOFF rather than BLP.

Using NOEOV, CMS does not do any end-of-tape processing on output. If this option is not specified, CMS writes a tape mark after it encounters EOT on output and, for SL and SUL files, also writes an EOVI label and another tape mark after the first tape mark. The tape is then rewound and unloaded. NOEOV suppresses this limited EOVI processing. In OS simulation, if you specify the NOEOV option, it is ignored during end-of-volume processing.

Using the FILEDEF ALT Operand: Use the ALT option for standard label tape files only. You can use alternate tape drive support as follows:

```
filedef filea tap1 sl (alt tap2
```

where tap2 is the alternate drive where the second volume of the tape file is mounted.

Responses

If FILEDEF is entered with no operands and there are no filedefs in effect, the message:

```
DMSFLD324I      No user defined FILEDEFS in effect
is displayed.
```

If FILEDEF is entered with no operands and there are filedefs in effect, a list of current definitions is displayed. For example:

```
ddname1 device1
.
.
.
ddnameN deviceN
```

if the device is a disk, additional file ID information is displayed:

```
ddname DISK   filename filetype filemode [datasetname]
```

or, if the device is a tape, additional label information is displayed:

```
ddname TAPn   labeltype [n [VOLID volid]][filename]
```

```
DMSFLD069I      Filemode mode not accessed
```

The specified disk or directory is not accessed; the file definition remains in effect. You should access the disk or directory before you attempt to read or write the file.

```
DMSFLD220R      Enter data set name:
```

A FILEDEF command with the DSN ? operand was entered. Enter the exact OS or VSE file identification, including embedded periods and blanks.

```
DMSFLD704I      Invalid CLEAR request
```

A CLEAR request was entered for a file definition that does not exist; no action is taken.

```
DMSSTT228I      User labels bypassed on data set data set name
```

This message is displayed when you issue a FILEDEF command for an OS data set that contains user labels. The message is displayed the first time you issue the FILEDEF command after accessing the disk on which the data set resides.

Messages and Return Codes

```
DMSFLD003E      Invalid option: option [RC = 24]
DMSFLD023E      No filetype specified [RC = 24]
DMSFLD027E      Invalid device devtype [for SYSaaa] [RC = 24]
DMSFLD029E      Invalid parameter parameter in the option option field [RC = 24]
DMSFLD035E      Invalid tape mode [RC = 24]
DMSFLD050E      Parameter missing after DDNAME [RC = 24]
```

FILEDEF

DMSFLD065E *option* option specified twice [RC=24]
DMSFLD066E *option1* and *option2* are conflicting options [RC=24]
DMSFLD070E Invalid parameter *parameter* [RC=24]
DMSFLD109S Virtual storage capacity exceeded [RC=104]
DMSFLD221E Invalid data set name [RC=24]
DMSFLD224E Fileid already in use [RC=24]
DMSFLD420E NSL exit filename missing or invalid [RC=24]
DMSFLD699E No filetype specified or *vdev* is an invalid disk address [RC=24]
DMSFLD735E Primary and alternate tape drives are identical. [RC=24]
DMSFLO447E Invalid sysparm information [RC=24]

FILELIST

Use the FILELIST command to display a list of information about CMS files residing on accessed disks and Shared File System (SFS) directories. For directories, its subdirectories are also listed. The LISTFILE and FILELIST commands display identical information, but in the FILELIST environment, information is displayed under the control of the System Product Editor. You can use XEDIT subcommands to manipulate the list itself. You can also issue CMS commands against the files directly from the displayed list.

Format

FILEList	[<i>fn</i> [<i>ft</i> [<i>fm</i>]]]	[(options... [])]
	Options: [Append] [Filelist] [PROFile <i>fn</i>]	[Nofilelist]
	[ALLfile] [STAts]	[SHAre]
	[AUPHfile]	[SEArch]

Operands

fn

is the name of the file or subdirectory for which information is to be collected. If an asterisk (*) is coded in this field, all file names are used.

Certain special characters (* and %) can be used as part of the file name to request that the list contain a specific subset of files. See "Pattern Matching" on page 8 for more information on using these characters.

ft

is the file type of the file(s) for which information is to be collected. If an asterisk is coded in this field, all file types are used.

Certain special characters can be used as part of the file type to request that the list contain a specific subset of files. See "Pattern Matching" on page 8 for more information on using these characters.

fm

is the file mode of the file(s) for which information is to be collected. If this field is omitted, only the disk or directory accessed as A is searched. If an asterisk is coded, all accessed disks and directories are searched.

If you not specify any operands, the list contains all files that are on your A disk, or in Shared File System (SFS) directories accessed as A, or SFS subdirectories accessed as A. Issuing FILELIST with no operands is like issuing "filelist * * a."

Options

Append

specifies that the list of files should be appended to the existing list. This option is meaningful only when issued from within the FILELIST environment. If issued outside of FILELIST, it results in an error condition.

The type of information (STATS, SHARE, or SEARCH) to be appended must match the existing type. You can display SHARE information by entering,
 filelist (share

To specify any appends, you must also specify the type of filelist you currently have displayed; such as,

filelist * * c (share append

Filelist

specifies that *fn ft fm* is a file that already contains a list of files, produced by an earlier invocation of FILELIST or LISTFILE (using the EXEC option). Information about each file in this list is displayed.

If this option is specified, no special characters used for pattern matching may appear in *fn ft* or *fm*. For information on pattern matching, see "Pattern Matching" on page 8.

For information on creating and saving a list of files, see the usage note, "Saving a List of Files."

Nofilelist

specifies that *fn ft fm* is not a list of files.

PROFile *fn*

specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the FILELIST command. If not specified, a macro named PROFFLST XEDIT is invoked. For more information on the PROFFLST macro, see the usage note, "Default PF Key Settings."

ALLfile

specifies that all of the files in an SFS directory be listed whether or not you have read or write authority to them. Subdirectories and erased or revoked aliases are also listed. ALLFILE is the default when the specified file mode refers to a directory, but this option is ignored when the file mode refers to a disk.

AUTHfile

specifies that only the files in an SFS directory that you have read or write authority to are to be listed. AUTHFILE is ignored if the file mode refers to a disk. Subdirectories and erased or revoked aliases are *not* listed.

STATs

lists the following information about the specified files:

- file identifier or directory identifier
- format and logical record length of the file
- number of records and number of blocks in the file
- date and time of last update.

See the "Examples" on page 212 for a sample display using the STATS option.

SHAre

lists the following information:

- file identifier or SFS subdirectory name

- file owner or SFS directory owner
- the type of the file or SFS directory (mdisk, directory, alias, base, erased or revoked alias)
- what authority you have to the file or directory (R/W).

The SHARE option can be used for files on a disk or in a directory. See the “Examples” for a sample display using the SHARE option.

SEArch

searches a directory structure for the specified file(s). This option is useful if you know a file name but not the directory that the file is in. The SEARCH option can only be used for files in a directory. The following information is listed:

- file identifier
- directory containing the file

See the “Examples” for a sample display using the SEARCH option.

The search begins with the directory that you specify as *fm*, and continues within that directory structure for all subdirectories for which you have read or write authority, whether they are accessed or not. Subdirectories that are locked EXCLUSIVE by another user are not searched. If *fm* is not specified the search begins at your top directory whether or not it is accessed. You cannot specify an asterisk (*) for the file mode when you use this option.

Usage Notes

1. Tailoring the FILELIST Command Options

You can use the DEFAULTS command to set up options and/or override command defaults for FILELIST. However, the options you specify on the command line when entering the FILELIST command override those specified in the DEFAULTS command. This allows you to customize the defaults of the FILELIST command, yet override them when you desire. Refer to the DEFAULTS command description for more information.

2. XEDIT Environment

When you invoke the FILELIST command you are placed in the XEDIT environment, editing a file “userid FILELIST A0.” A sample FILELIST screen is shown in the “Examples” section.

The full power of XEDIT is available to you while you issue commands against the list of files. For example, you may want to use XEDIT subcommands to scroll through the list of files, locate a particular file, etc.

However, some XEDIT subcommands are inappropriate in this environment. Subcommands that alter the format or the contents of “userid FILELIST” (for example, SET LRECL, SET TRUNC, SET FTYPE, or SET LINEND) may cause unpredictable results.

3. Entering CMS commands from FILELIST

Begin CMS commands with “CMS” to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.

4. If you want to issue FILELIST from an exec program, you should precede it with the EXEC command; that is, specify

```
exec filelist
```

5. File Sharing Considerations

While you have your FILELIST displayed on the screen, other users may modify the files (if you have authorized them to do so). As these changes are saved or filed, some of the file attributes on your screen may be out of date. When you issue a command against such a file, you may receive a message:

```
DMSWEX654E Invalid symbol symbol; {/0 must be specified
           alone|invalid character char following
           / symbol} [RC=24]
```

If you do receive this message, simply clear the rest of the line following your command and press enter.

6. Saving a List of Files

You can save a list of files created by the FILELIST command simply by filing it, that is, issuing FILE or SAVE from the command line. Remember that the list is a file, whose file name is your user ID and whose file type is FILELIST. If you issue FILE or SAVE, the file "userid FILELIST" is kept until the next time you issue FILE or SAVE from the list.

You can also save a particular list of files by filing it under a different file ID. One way to do this is to issue the XEDIT subcommand FILE from the command line, specifying a different file name and/or file type. For example, you could issue "FILE MY FILES." Another way is to issue FILE from the command line, and then to use the RENAME command.

Saving a list of files is useful when you want to send multiple files using the SENDFILE command. The list of files that you saved can be specified in the SENDFILE command issued with the FILELIST option. With this method, you can send multiple files by issuing the SENDFILE command only once. The only file identifier you have to keep track of is that of the list. For information on sending a list of files, see the SENDFILE command (the description of the Filelist option).

7. Issuing Commands From the List

On a full screen display, you can issue commands directly from the line on which a file is displayed. You do this by moving the cursor to the line that describes the file, typing the command in the space provided to the left of the file name, and then pressing the ENTER key to execute the command.

If a command is longer than the command space provided on the screen, just continue typing over the information in the line. You may type over the entire line displayed, up to column 79.

When you press the ENTER key, all commands typed on one screen are executed, and the screen is restored to its previous state. However, the list is updated to reflect the current status of the files (see Response section).

You may want to enter commands from the FILELIST command line before executing commands that are typed on the list. To do this, move the cursor to the command line by using the PF12 key (instead of the ENTER key). After typing a command on the command line and pressing ENTER, you can use PF12 to move the cursor back to its previous position on the list.

8. You can use the special commands EXECUTE, DISCARD, ALIALIST, and AUTHLIST from the FILELIST screen. The EXECUTE command allows you to issue commands that use the files and directories displayed by FILELIST. The DISCARD command allows you to erase the files and directories displayed by FILELIST. The ALIALIST command displays alias information in a full-screen environment. The AUTHLIST command displays authority

information in a full-screen environment. For more information, see *Chapter 4 - Special Commands Used in Command Environments*.

9. Files with mixed-case file IDs can only be manipulated by user-written execs and modules. Files will have to be renamed in order to be used by EXECUTE.
10. Using Symbols as Part of a Command

Symbols can be used to represent operands in the command to be executed. They can be used in the commands typed on the screen, or as part of the command in EXECUTE (on the command line). Symbols are needed if the command to be executed has operands or options that follow the file ID. Examples of using symbols are in the "Examples" section, below.

The following symbols can be used:

- /** means one of three things:
 - a. If a file is displayed on the line, the slash (/) means file name, file type, file mode.
 - b. If a subdirectory is displayed on the line, the slash (/) means + *filemode.filename* for both the STATS and SHARE option screens.
 - c. If a subdirectory is displayed on the line, the slash (/) means file name, file type, directory name (or file mode if the subdirectory is accessed) for the SEARCH option screen.
- /n** means the file name displayed on the line.
- /t** means the file type displayed on the line.
- /m** means the file mode displayed on the line.
- /o** means execute the line as is, and omit appending anything.
- /d** means the directory name for the file. If file is on a minidisk, /d has same meaning as /m.

Any combinations of symbols can be used. For example:

- /n /t** means the file name followed by file type
- /nt** means the file name followed by file type
- /ntd** means the file name, file type, and directory name
- /tn** means the file type followed by file name
- /ntm** is equivalent to / alone
- /nnt** means file name followed by file name and file type.

Note: If the symbol '/' appears in a command or in its operands, it must be issued from the command line, and not as part of an EXECUTE command.

11. Special Symbols Used Alone

The following special symbols can be typed alone on the lines of the FILELIST display. They have the following meanings:

- =** means execute the previous command for this file. Commands are executed starting at the top of the screen. For example, suppose you enter the DISCARD command on the top line. You can then type an equal sign on any other line(s). Those files preceded by equal signs are discarded when the EXECUTE command is entered (from the command line or by pressing the ENTER key).

? means display the last command executed. The command is displayed on the line in which the ? is entered.

/ means make this line the current line. (On the FILELIST screen, the current line is the first file on the screen.)

12. Pattern Matching Subdirectory Names

If special characters (* or %) are used in *fn* or *ft* and the files are in an SFS directory, the specified directory cannot be open.

When pattern matching is done on subdirectory names that contain more than eight characters, the first eight characters are used as the file name and the remaining characters are used as the file type. For example, your directory accessed as A contains,

```
CROCKETT NOTEBOOK
CROCKETTNOTES
```

where CROCKETT NOTEBOOK is a file and CROCKETTNOTES is a subdirectory. Issuing,

```
filelist croc* n* a (allfile
```

would find both CROCKETT NOTEBOOK and CROCKETTNOTES because CROCKETTNOTES contains more than eight characters and is matched as if it had a file name of CROCKETT and a file type of NOTES. Issuing,

```
filelist crockettnotes * a (allfile
```

would also find both CROCKETT NOTEBOOK and CROCKETTNOTES because only the first eight characters are used as the file name. Therefore, any file or alias with the name of CROCKETT, or any subdirectory with CROCKETT as the first eight characters in its name would be listed.

13. Default Key Settings for STATS Option

Entering the FILELIST command with the STATS option executes the PROFFLST XEDIT macro, unless you specify a different macro as an option in the FILELIST command. If you use a PF key to switch back and forth between FILELIST screens, for example, the STATS and SHARE screen, the default option profiles are executed, even if you specified another profile in the initial FILELIST command. If you want to always use another profile, see the DEFAULTS command. Note that the setting of some keys depends on whether the file mode refers to a disk or directory. The keys are set to the following values by PROFFLST XEDIT:

Key	Setting	Disk or Directory	Action
ENTER	Execute	Both	Execute command(s) typed on file line(s) or on the command line.
PF 1	Help	Both	Display FILELIST command description.
PF 2	Refresh	Both	Update the list to indicate new files, erased files, etc., using the same parameters as those specified when FILELIST was invoked.

Key	Setting	Disk or Directory	Action
PF 3	Quit Quit	Disk Directory	Exit from FILELIST. Exit from FILELIST, or move up one level in the directory structure if you have used PF11 to display the contents of a subdirectory.
PF 4	Sort (type) Cancel	Disk Directory	Sort by file type, file name. Exit from FILELIST regardless of how many levels deep in the directory structure you currently are.
PF 5	Sort (date) Sort (dir)	Disk Directory	Sort by date and time, newest to oldest. Directories are listed by date and time, then files are listed by date and time. The SDIR XEDIT macro does this.
PF 6	Sort (size)	Both	Sort by size, largest first.
PF 7	Backward	Both	Scroll back one screen.
PF 8	Forward	Both	Scroll forward one screen.
PF 9	Fl/n	Both	Issue the command FILELIST /n * * at the cursor, so that a list is displayed, containing all files that have the file name that is displayed on the line containing the cursor (all file types and file modes).
PF 10	Share	Disk Directory	Not assigned. Issue a FILELIST command with the SHARE option on the same file(s) that was specified on the initial FILELIST command.
PF 11	XEDIT XED/FILEL	Disk Directory	Edit the file where the cursor is placed. If the cursor is on a line containing a file, edit the file where the cursor is placed. If the cursor is on a line with a directory, the contents of the directory are displayed in the FILELIST environment. If the directory is not already accessed, it is temporarily accessed as the last available file mode in the search order. Pressing PF3 from this screen will return you to the parent directory and the temporary file mode is automatically released. Using PF3 and PF11 moves you up and down your directory structure.
PF 12	Cursor	Both	If the cursor is in the file area, move it to the command line. If the cursor is on the command line, move it back to its previous location in the file (or to the current line).

Notes:

- a. If you specify an asterisk for the file mode, or use the FILELIST option,
 - PF 4 is set to Sort (type)
 - PF 5 is set to Sort (date)
 - PF 10 is set to Share
 - PF 11 is set to XED/FILEL

b. On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFFLST XEDIT macro sets synonyms that you can use to sort your FILELIST files. The synonyms are:

- SNAME** Sorts the list alphabetically by file name, file type, and file mode.
- STYPE** Sorts the list alphabetically by file type, file name, and file mode.
- SMODE** Sorts the list by file mode, file name, and file type.
- SRECF** Sorts the list by record format, file name, file type, and file mode.
- SLREC** Sorts the list by logical record length and then by size (greatest to least).
- SSIZE** Sorts the list by number of blocks and number of records (greatest to least).
- SDATE** Sorts the list by year, month, day, and time (most recent to oldest).
- SDIR** Sorts the directories by date and time, then files are listed by date and time. SDIR is an XEDIT macro.

14. Default Key Settings for SHARE Option

Entering the FILELIST command with the SHARE option executes the PROFFSHR XEDIT macro, unless you specify a different macro as an option in the FILELIST command. Note that the setting of some keys depends on whether the file mode refers to a disk or directory. The keys are set to the following values by PROFFSHR XEDIT:

Key	Setting	Disk or Directory	Action
ENTER	Execute	Both	The same as for the STATS option.
PF 1	Help	Both	The same as for the STATS option.
PF 2	Refresh	Both	The same as for the STATS option.
PF 3	Quit	Disk	The same as for the STATS option.
	Quit	Directory	The same as for the STATS option.
PF 4	Sort (type)	Disk	The same as for the STATS option.
	Cancel	Directory	The same as for the STATS option.
PF 5	Sort (date)	Disk	The same as for the STATS option.
	Sort (dir)	Directory	The same as for the STATS option.
PF 6	Sort (size)	Disk	The same as for the STATS option.
	Auth	Directory	Issue an AUTHLIST command for the file on the line containing the cursor.
PF 7	Backward	Both	The same as for the STATS option.
PF 8	Forward	Both	The same as for the STATS option.
PF 9	Fl/n	Disk	The same as for the STATS option.
	Alias	Directory	Issue an ALIALIST command for the file on the line containing the cursor.

Key	Setting	Disk or Directory	Action
PF 10	Stats	Disk Directory	Not assigned. Issue a FILELIST command with STATS option on the same file(s) that was specified on the initial FILELIST command.
PF 11	XEDIT XED/FILEL	Disk Directory	The same as for the STATS option. The same as for the STATS option.
PF 12	Cursor	Both	The same as for the STATS option.

Notes:

a. If you specify an asterisk for the file mode, or use the FILELIST option,

PF 4 is set to Sort (type)

PF 5 is set to Sort (date)

PF 6 is set to Auth

PF 9 is set to Alias

PF 10 is set to Stats

PF 11 is set to XED/FILEL

b. On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFFSHR XEDIT macro sets synonyms that you can use to sort your FILELIST screen. The synonyms are:

SNAME Sorts the list alphabetically by file name, file type, and file mode.

STYPE Sorts the list alphabetically by file type, file name, and file mode.

SMODE Sorts the list by file mode, file name, and file type.

SOWNER Sorts the list by owner, file name, file type, and file mode.

STYP Sorts the list by type, file name, file type, and file mode.

SWRITE Sorts the list by W.

SDATE Sorts the list by year, month, day, and time (most recent to oldest).

SSIZE Sorts the list by number of blocks and number of records (greatest to least).

SDIR Sorts the directories by date and time, then files are listed by date and time. SDIR is an XEDIT macro.

15. Default Key Settings for SEARCH Option

Entering the FILELIST command with the SEARCH option executes the PROFFSEA XEDIT macro, unless you specify a different macro as an option in the FILELIST command. It sets the keys to the following values (note that the SEARCH option is only valid for files in SFS directories):

FILELIST

Key	Setting	Disk or Directory	Action
ENTER	Execute	Directory	The same as for the STATS option.
PF 1	Help	Directory	The same as for the STATS option.
PF 2	Refresh	Directory	The same as for the STATS option.
PF 3	Quit	Directory	The same as for the STATS option.
PF 4	Dirlist	Directory	Issues a DIRLIST /d command for the directory on the line containing the cursor.
PF 5	Sort (name)	Directory	Sorts the list of files by file name, file mode.
PF 6	Auth	Directory	Issues an AUTHLIST command for the file on the line containing the cursor.
PF 7	Backward	Directory	The same as for the STATS option.
PF 8	Forward	Directory	The same as for the STATS option.
PF 9	Alias	Directory	Issues an ALIALIST command for the file on the line containing the cursor.
PF 10	Filelist	Directory	Issues a 'FILELIST * * /m' command for the directory on the line containing the cursor. If the directory is not accessed, a temporary access is done using the last available file mode in the search order. Use the PF 3 or PF 4 key to return to the display of SEARCH information.
PF 11	XEDIT	Directory	Same as for the STATS option.
PF 12	Cursor	Directory	The same as for the STATS option.

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFFSEA XEDIT macro sets synonyms that you can use to sort your FILELIST screen. The synonyms are:

SNAME Sorts the list alphabetically by file name, file type, and file mode.

STYPE Sorts the list alphabetically by file type, file name, and file mode.

SMODE Sorts the list by file mode, file name, and file type.

SDIR Sorts the list by directory name, file name, and file type.

Examples

For the examples in this section, the *fm* is portrayed as an SFS directory. Sample screens are shown in this section for the SEARCH, SHARE, and STATS options.

The following FILELIST screen was created by issuing the FILELIST command with no operands - which is equivalent to entering `filelist * * A (stats`. Note that the files are sorted by date and time, newest to oldest.

```

GRASSI FILELIST A0 V 108 Trunc=108 Size=11 Line=1 Col=1 Alt=0
Directory = SERVER1:GRASSI.GOODIES.FOOD
Cmd Filename Filetype Fm Format Lrec1 Records Blocks Date Time
PIZZA TOPPINGS A1 F 107 281 10 10/04/88 17:59:00
COOKIE ASSEMBLE A1 F 98 49 2 10/03/88 15:17:01
JELLY BEANS A1 F 120 277 10 9/25/88 9:14:02
DIETING TIPS A1 F 75 28 1 9/24/88 12:10:03
CUSTOMER LIST A1 F 95 34 2 8/04/88 21:12:04
SALES A DIR - - - 8/04/88 14:34:34
SEND EXEC A1 F 80 101 4 8/04/88 15:33:05
INVENTORIES A DIR - - - 8/01/88 16:50:06
MYMACRO XEDIT A1 V 95 29 2 7/30/88 20:58:07
CMSFILES SCRIPT A1 V 80 489 30 7/26/88 16:05:08
JUNK FOOD A1 - - - - - -

```

1= Help 2= Refresh 3= Quit 4= Cancel 5= Sort(dir) 6= Sort(size)
7= Backward 8= Forward 9= FL /n 10= Share 11= XED/FILEL 12= Cursor

====>

X E D I T 1 File

Figure 5. Sample FILELIST Screen with STATS option

See "Responses" on page 216 for an explanation of the information listed in each column.

The following FILELIST screen was created by issuing:

```
filelist * * a (share
```


FILELIST

```
GRASSI FILELIST A0 V 149 Trunc=149 Size=11 Line=1 Col=1 Alt=0
Directory = SERVER1:GRASSI.GOODIES.FOOD
Cmd Filename Filetype Fm Owner Type R W
PIZZA TOPPINGS A1 REYNOLDS ALIAS X -
COOKIE ASSEMBLE A1 CROCKETT ALIAS X X
JELLY BEANS A1 STONE ALIAS X -
DIETING TIPS A1 GRASSI BASE X X
CUSTOMER LIST A1 HALL ALIAS X -
SALES A GRASSI DIR X X
SEND EXEC A1 GRASSI BASE P P
INVENTORIES A GRASSI DIR X X
MYMACRO XEDIT A1 GRASSI BASE X X
CMSFILES SCRIPT A1 GRASSI BASE X X
JUNK FOOD A1 HALL ERASED - -

1= Help 2= Refresh 3= Quit 4= Cancel 5= Sort(dir) 6= Auth
7= Backward 8= Forward 9= Alias 10= Stats 11= XED/FILEL 12= Cursor

====>

X E D I T 1 File
```

Figure 6. Sample FILELIST Screen with SHARE option

See "Responses" on page 216 for an explanation of the information listed in each column.

The following FILELIST screen was created by issuing:

```
filelist * * a (search
```

```

GRASSI FILELIST A0 V 355 Trunc=355 Size=13 Line=1 Col=1 Alt=0
Cmd Filename Filetype Fm Directory Name
CMSFILES SCRIPT A1 SERVER1:GRASSI.GOODIES.FOOD
COOKIE ASSEMBLE A1 SERVER1:GRASSI.GOODIES.FOOD
CUSTOMER LIST A1 SERVER1:GRASSI.GOODIES.FOOD
DIETING TIPS A1 SERVER1:GRASSI.GOODIES.FOOD
JELLY BEANS A1 SERVER1:GRASSI.GOODIES.FOOD
JUNK FOOD A1 SERVER1:GRASSI.GOODIES.FOOD
MYMACRO XEDIT A1 SERVER1:GRASSI.GOODIES.FOOD
PIZZA TOPPINGS A1 SERVER1:GRASSI.GOODIES.FOOD
SEND EXEC A1 SERVER1:GRASSI.GOODIES.FOOD
STOCK LIST - SERVER1:GRASSI.INVENTORIES.
84 LIST B1 SERVER1:GRASSI.INVENTORIES.PRICES
85 LIST B1 SERVER1:GRASSI.INVENTORIES.PRICES
SENDIT EXEC - SERVER1:GRASSI.SALES

1= Help 2= Refresh 3= Quit 4= Dirlist 5= Sort(name) 6= Auth
7= Backward 8= Forward 9= Alias 10= Filelist 11= XEDIT 12= Cursor

====>
XEDIT 1 File

```

Figure 7. Sample FILELIST Screen with SEARCH option

See “Responses” on page 216 for an explanation of the information listed in each column.

Examples of Using Symbols: The following examples show how symbols can be used to represent operands in a command. The values substituted for the symbols and the resulting command are shown. In each case, the command can be entered in either of the following ways:

- typed in the “Cmd” area of the screen. The command is executed either by entering EXECUTE on the XEDIT command line and then pressing ENTER, or simply by pressing ENTER.
- entered from the XEDIT command line, as an operand of EXECUTE (in the form “EXECUTE lines command”).

If a symbol is not specified, the file name, file type, and file mode are appended automatically to the command.

FILE ID	COMMAND	RESULTING COMMAND
pizza toppings a	discard	discard pizza toppings a
cookie assemble a	assemble /n	assemble cookie
jelly beans a	copy / = flavors =	copy jelly beans a jelly flavors a
dieting tips a	copy / /nt b	copy dieting tips a dieting tips b

FILELIST

Responses

Issuing the FILELIST command with the STATS option displays the following information:

Filename	Filetype	Fm	Format	Lrecl	Records	Blocks	Date	Time
fn	ft	fm	format	lrecl	norecs	noblks	mm/dd/yy	hh:mm:ss
.
.
.

where:

fn

is the name of the file or directory.

ft

is the file type of the file. For a directory this column is blank.

fm

is the file mode of the disk or parent directory.

format

is the format: F is fixed-length, V is variable-length, DIR is a directory. A dash indicates an erased or revoked alias.

lrecl

is the logical record length of the largest record in the file. For directories and revoked or erased aliases, a dash is displayed.

norecs

is the number of logical records in the file. For directories and revoked or erased aliases, a dash is displayed.

noblks

is the number of CMS data blocks that the file occupies. For directories and revoked or erased aliases, a dash is displayed.

mm/dd/yy

is the date (month/day/year) that the file was last updated. For a directory, the date the directory was created is displayed. A dash appears in this column for erased or revoked aliases.

hh:mm:ss

is the time (hours:minutes:seconds) that the file was last updated. For a directory, the time the directory was created is displayed. A dash appears in this column for erased or revoked aliases.

One entry is displayed for each file or subdirectory listed.

If the SHARE option is specified, the information displayed is:

Filename	Filetype	Fm	Owner	Type	R	W
fn	ft	fm	owner	type	r	w
.
.
.

where:

owner

is the user ID of base file owner. For files on disks, this will be the disk owner's user ID.

type

is one of the following:

- MDISK for a file on a disk
- BASE for a base file in a directory
- DIR for a directory
- ALIAS for an alias in a directory
- ERASED for an erased alias
- REVOKED for a revoked alias

r

is read authority.

X indicates that you have read authority.

— A dash indicates that you do not have read authority.

P means that the authority is managed by an External Security Manager (ESM).

For files on disk, an X will be displayed if the disk is accessed read/only or read/write.

w

is write authority.

X indicates that you have write authority.

— A dash indicates that you do not have write authority.

P means that the authority is managed by an External Security Manager (ESM).

For files on disk, an X will be displayed if the disk is accessed read/write, a dash if accessed read/only.

One entry is displayed for each file or subdirectory listed.

If the SEARCH option is specified, the information displayed is:

Filename	Filetype	Fm	Directory
fn	ft	fm	directory
.	.	.	.
.	.	.	.
.	.	.	.

where:

directory

is the complete name of directory that contains the file.

One entry is displayed for each file listed. If the specified files are on a disk, you will receive the message:

DMSWFL1182E The SEARCH option may not be used with a minidisk

When a command is executed, one of the following symbols is displayed in the "Cmd" space to the left of the file for which the command was executed.

- * Means the command was executed successfully (RC=0).
- *n Is the return code from the command executed (RC=n).
- *? Means that the command was an unknown CP/CMS command (RC=-3).

FILELIST

*! Means that the command was not valid in CMS subset. For a list of commands valid in CMS subset mode, see the *VM/SP CMS User's Guide*.

The following responses can also appear directly on the FILELIST screen:

```
*   fname  ftype  fmode  ** Not found. **
*   No files match the search criteria: fname ftype fmode
*   fname  ftype  fmode  ** Discarded, Renamed, or Relocated **
*   fname  ftype  fmode  ** Fileid is in Mixed Case.
                               Invalid for EXECUTE *
*   fname  ftype  fmode  has been discarded.
File fname ftype fmode has been discarded.
```

Messages and Return Codes

DMSWFL002E File *fn ft fm* not found [RC=28]
DMSWFL054E Incomplete fileid specified [RC=24]
DMSWFL651E APPEND must be issued from RDRLIST or FILELIST [RC=40]
DMSWEX654E Invalid symbol *symbol*; {/0 must be specified alone|invalid character *char* following / symbol [RC=24]
DMSWFL653E Error executing LISTFILE, rc=*nn* [RC=*nn*]
DMSWFL680E Invalid fileid specified with FILELIST option [RC=20]
DMSWFL1067E Return code *nn* from the CMS XEDIT command [RC=00]
DMSWFL1182E The SEARCH option may not be used with a minidisk [RC=74]
DMSWFL1183E '*' may not be specified for the filemode with the SEARCH option [RC=24]
DMSWFL1223E There is no default file pool currently defined [RC=40]
DMSWFL1227E No filemode is available to access directory [RC=00]
DMSWFL1228E Error executing ACCESS for directory, rc=*nn* [RC=*nn*]
DMSWFL1229E Directory is empty [RC=00]
DMSWFL1232E SDIR must be issued from FILELIST Share or Stats screen [RC=40]
DMSWFL1233E Invalid use of APPEND option [RC=40]
DMSWFL1234E Error executing FILELIST, rc=*nn* [RC=*nn*]
DMSWFL1249I Directory has been temporarily accessed as filemode *fm* [RC=00]
DMSWFL1263E You are not authorized for directory [RC=00]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813

FINIS

Use the FINIS command to close one or more files on a disk or in a Shared File System (SFS) directory.

Format

FINIS	<i>fn</i> <i>ft</i> [<i>fm</i>] * * [*]
--------------	--

Operands

fn
is the file name of the file to be closed. If you code an asterisk (*) in this field, all file names with the specified file type, and file mode are closed.

ft
is the file type of the file to be closed. If you code an asterisk (*) in this field, all files with the specified file name, and file mode for all file types are closed.

fm
is the file mode of the file to be closed. If you code an asterisk (*) in this field, all files with the specified file name and file type are closed. If this field is omitted, an asterisk is assumed.

Usage Notes

1. If the specified file(s) reside on a minidisk, no changes are committed to the minidisk until all files on that minidisk that are open for output are closed.
2. If the specified file(s) reside in an SFS directory, the FINIS command will commit any changes. However the commit will not take effect if there are any other files or directories opened on the same work unit.
3. The FINIS command closes files opened by the FSOPEN macro, (or the EXECIO command within execs). Files opened by the CSL routine, OPEN, must be closed by the CSL routine, CLOSE.

Messages and Return Codes

DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC = 31]
DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813

If an error occurs, register 15 contains one of the following error codes:

Code	Meaning
6	File not open (or no read or write was issued to the file), or invalid file ID (fn ft fm) specified.
31	Rollback occurred for this work unit due to an error in close.

FORMAT

Use the FORMAT command to:

- Initialize a minidisk for use with CMS files
- Count or reset the number of cylinders on a minidisk
- Write a label on a minidisk

Format

FORMAT	<p><i>vdev</i> <i>fm</i> [<i>nocyl</i>] [(options...[])]</p> <p>[<i>noblk</i>]</p> <p>Options:</p> <table border="1"> <tr> <td style="text-align: center;"><u>Blksize</u></td> <td style="text-align: center;"> 512 800 1024 2048 4096 1K 2K 4K </td> <td style="text-align: center;"> [Noerase] [Label] [Recomp] </td> </tr> </table>	<u>Blksize</u>	512 800 1024 2048 4096 1K 2K 4K	[Noerase] [Label] [Recomp]
<u>Blksize</u>	512 800 1024 2048 4096 1K 2K 4K	[Noerase] [Label] [Recomp]		

Operands

vdev

is the virtual device address of the minidisk to be formatted.

Valid addresses are:

- 0001 through FFFF for a 370-XA mode virtual machine
- 001 through 5FF for a VM/SP virtual machine in basic control mode
- 001 through FFF for a System/370 mode virtual machine and VM/SP virtual machine in extended control mode.

On VM/SP and System/370 mode virtual machines you can supply a leading zero.

Note: In the preceding description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. Valid addresses in both environments are described to help you plan and develop applications that will run in both environments.

fm

is the file mode letter to be assigned to the specified device address. Any single character letter, A through Z, is valid. This field must be specified. If any other disk is accessed at this mode, it is released.

nocyl

is the number of cylinders to be made available for use. All available cylinders on the disk are used if the number specified exceeds the actual number available.

noblk

is the number of FB-512 blocks to be made available for use. If the number specified exceeds the actual number of blocks on the disk, then all the blocks on the disk are made available for use.

Options

Blksize

specifies the physical DASD block size of the CMS minidisk. The block sizes 1024, 2048, and 4096 may alternately be specified as 1K, 2K, and 4K, respectively. For FB-512 devices, only block sizes 512, 1024, 2048, and 4096 are supported; for CKD (count key data) devices, all block sizes are supported.

The **BLKSIZE** option defaults to a block size that optimizes the I/O and data storage for the particular device. CKD devices default as follows:

DASD	Default Blocksize
2314	1024
3340	1024
3330	2048
3350	2048
3375	4096
3380	4096

FB-512 devices, such as 9313, 9332, 9335, 3310, and 3370, default to a block size of 1024. For more information on choosing an appropriate blocksize, see Usage Note 7.

Noerase

specifies for FB-512 devices that the permanently formatted FB-512 blocks are not to be cleared to zeros. If not specified, the FB-512 blocks will be cleared. For non-FB-512 devices, this option is ignored.

Label

writes a label on the disk without formatting the disk. The CMS disk label is written on cylinder 0, track 0, record 3 of the minidisk or block1 of an FB-512 device. A prompting message requests a six-character disk label (fewer than six characters are left-justified and blanks padded).

Recomp

changes the number of cylinders/blocks (FB-512 blocks) on the disk that are available to you. If you specify NOCYL/NOBLK and there are not that many cylinders/blocks available, you only get the available number of cylinders/blocks. For 800-byte blocks, all cylinders are used. If you do not specify NOCYL/NOBLK, or if NOCYL/NOBLK is being increased for a disk formatted in 512-, 1K-, 2K-, or 4K-byte blocks, the maximum number of cylinders or FB-512 blocks *last* formatted on the disk is made available to you.

Usage Notes

1. You can use the **FORMAT** command with any virtual 9313, 9332, 9335, 3310, 3330, 3340, 3350, 3370, 3375, 3380, or 2314 device. The speed matching buffer feature (Feature #6550) for the 3380 supports the use of extended count-key-data channel programs.

Note: The speed matching buffer is not supported for 3380 Models AD4/BD4 or AE4/BE4.

- When you do not specify either the RECOMP or LABEL option, the disk area is initialized by writing a device-dependent number of records (containing binary zeros) on each track. Any previous data on the disk is erased. A read after write check is made as the disk is formatted. For example:

```
format 191 a 25
```

initializes 25 cylinders of the disk located at virtual address 191 in CMS format. The command:

```
format 192 b 25 (recomp)
```

changes the number of cylinders available at virtual address 192 to 25 cylinders, but does not erase any existing CMS files. To change only the label on a disk, you can enter:

```
format 193 c (label)
```

Respond to the prompting message with a six-character label.

- If you want to format a minidisk for VSAM files, you must use the Device Support Facility. If you want to format an entire disk, you may use any OS or DOS disk initialization program.
- Because the FORMAT command requires heavy processor utilization and is heavily I/O bound, system performance may be degraded if there are many users on the system when you use FORMAT.
- When formatting FB-512 devices, enough blocks of the minidisk area must be formatted to support the CMS disk structure, or message DMS216E will be displayed, and the FORMAT request will be terminated. The number of FB-512 blocks which must be formatted for minidisks of 512-, 1K-, 2K-, and 4K-byte CMS blocksize is 6, 12, 24, and 48, respectively.
- If the FORMAT command with the RECOMP option fails and CMS issues message DMSFOR214W, "CANNOT RECOMPUTE WITHOUT LOSS OF DATA. NO CHANGE.," query your A-disk to determine the number of unallocated cylinders. If the number of cylinders seems adequate, it is possible that some of the allocated space is at the end of the disk, and is thus not available to the FORMAT command. Issue the command:

```
COPY * * A = = = (REP
```

followed by the FORMAT command with the RECOMP option.

- Choosing an appropriate BLKSIZE to format a disk depends upon its intended use. A 4K BLKSIZE will optimize the I/O if the disk is to contain large files with no missing records (dense). A BLKSIZE of 1K is more appropriate when creating many small files or sparse files. For example, PL/I regional files are sparse and they may allocate more space on a 4K disk than on a 1K disk, therefore, the smaller BLKSIZE is preferable.

The larger the block size of the disk, the greater the amount of storage required for input/output buffers. Each buffer that the system needs must be a contiguous block of system keyed storage. The size of this area of storage being the block size of the disk. Programs that dynamically allocate storage based upon machine size may use up all of the available storage. This may not allow the system enough storage to allocate buffers for its use. Consequently, a program needing a 4K disk that uses all of the available storage may be unable to get I/O buffers if they are not already allocated. For more information on CMS storage management, refer to the *VM/SP CMS Diagnosis Reference*.

8. Because the CMS file system uses a five level tree structure when using the 512-byte block size, the maximum number of data blocks for a variable format file is about 15 times less than the actual limit (2 to the power of 31 minus 1).
9. A CMS nucleus cannot be saved on a CKD device formatted with 512-byte block size.

Responses

DMSFOR603R Format will erase all files on disk *mode(vdev)*. Do you wish to continue? Enter 1 (YES) or 0 (NO).

To reply yes, enter 1 or 'YES'. To reply no, enter 0 or 'NO'. If you respond 'YES', you must *only* enter the character string 'YES'. You have indicated that a disk area is to be initialized; all existing files are erased. If the character string contains leading or trailing blanks, such as ' YES' or 'YES ', the response is processed as a 'NO' response. Responding 'NO', pressing the ENTER key, or entering a character string other than 'YES' cancels execution of the FORMAT command.

DMSFOR605R Enter disk label:

You have requested that a label be written on the disk. Enter a one- to six-character label.

DMSFOR705I Disk remains unchanged

The response to message DMSFOR603R was other than 'YES'.

DMSFOR732I *nnnn* {cylinders|FB-512 blocks} formatted on *mode(vdev)*

The format operation is complete.

DMSFOR733I Formatting disk *mode*

The disk represented by mode letter 'mode' is being formatted.

LABEL	CUU	M	STAT	CYL	TYPE	BLKSIZE	FILES	BLKS	USED-(%)	BLKS	LEFT	BLK	TOTAL
label	cuu	m	R/W	nnnn	type	blksize	nnnnn	nnnn-	%	nnn	nnnnn		

This message provides the status of a disk when you use the RECOMP option. The response is the same as when you issue the QUERY command with the DISK operand.

Messages and Return Codes

DMSFOR003E	Invalid option: <i>option</i> [RC=24]
DMSFOR005E	No BLKSIZE specified [RC=24]
DMSFOR017E	Invalid device address <i>vdev</i> [RC=24]
DMSFOR028E	No device specified [RC=24]
DMSFOR037E	Filemode <i>mode[(vdev)]</i> is accessed as read/only [RC=36]
DMSFOR048E	Invalid mode <i>mode</i> [RC=24]
DMSFOR069E	Filemode <i>mode</i> not accessed [RC=36]
DMSFOR070E	Invalid parameter <i>parameter</i> [RC=24]
DMSFOR113S	Device <i>vdev</i> not attached [RC=100]
DMSFOR114S	Device <i>vdev</i> is an unsupported device type, or requested BLKSIZE is not supported for the device [RC=88]
DMSFOR125S	Permanent unit check on disk <i>mode(vdev)</i> [RC=100]
DMSFOR126S	Error {reading writing} label on disk <i>mode(vdev)</i> [RC=100]
DMSFOR214W	Cannot recompute without loss of data; no change [RC=8]
DMSFOR216E	Insufficient blocks on disk to support CMS disk structure [RC=100]

FORMAT

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

GENDIRT

Use the GENDIRT command to fill in a CMS auxiliary directory. The auxiliary directory contains the name and location of modules that would otherwise significantly increase the size of the resident directory, thus increasing search time and storage requirements. By using GENDIRT to fill in an auxiliary directory, the file entries for the given command are loaded only when the command is invoked.

Format

GENDIRT	<i>directoryname</i> [<i>targetmode</i> [<i>sourcemode</i>]]
----------------	---

Operands

directoryname

is the entry point of the auxiliary directory.

targetmode

is the file mode letter of the disk containing the modules referred to in the directory. The letter is the file mode of the disk containing the modules at execution time, not the file mode of the disk at creation of the directory. At directory creation time, all modules named in the directory being created must be on either the A-disk or a read-only extension; that is, not all disks are searched. The default value for *targetmode* is S (system disk). It is your responsibility to determine the usefulness of this operand at your installation, and to inform all users whose programs are in auxiliary directories exactly what file mode to specify on the ACCESS command.

sourcemode

is the mode of the disk that contains the modules or files when the GENDIRT command is issued. If not specified, 'A' is the default.

Note: For information on creating auxiliary directories and for further requirements for using the *targetmode* option, see the *VM/SP Application Development Guide for CMS*.

Messages and Return Codes

DMSGND002W File *fn ft [fm]* not found [RC=4 or 28]
 DMSGND021E Entry point *name* not found [RC=40]
 DMSGND022E No directory name specified [RC=24 or 28]
 DMSGND070E Invalid parameter *parameter* [RC=24]
 DMSGND1211W FST for file *fn ft fm* not copied [RC=4]
 DMSGND1264E Filemode *fm* is not associated with a minidisk [RC=4]

GENMOD

Use the GENMOD command to create both relocatable and non-relocatable MODULE files on a disk or directory.

The GENMOD command behaves differently depending upon the environment it is issued from. In the following description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. The behavior of the GENMOD command in both environments is described below to help you plan and develop applications that will run in both environments.

The GENMOD command will save the AMODE and RMODE attributes in the file. If the module has architectural dependencies, you can give the module an architectural attribute to restrict its execution to that particular type of architecture.

The GENMOD command is the final step in the CMS link edit process, which also includes the LOAD and INCLUDE commands.

You can load a MODULE file created by the GENMOD command by using the:

- LOADMOD command
- NUCXLOAD command.

To execute a module that has been loaded with the LOADMOD command, issue the START command.

To load and execute a MODULE file in a single step, invoke the file as a command by using its file name as the command name.

Format

Genmod	<pre>[fn [MODULE [fm]]][(options...[])] [A1]</pre> <p>Options: [FROM entry 1][TO entry 2]</p> <table style="margin-left: 40px; border: none;"> <tr> <td>[MAP NOMAP]</td> <td>[STR NOSTR]</td> <td>[OS DOS ALL]</td> </tr> <tr> <td>[CLEAN NOCLEAN]</td> <td>[SYSTEM]</td> <td></td> </tr> <tr> <td>[AMODE 24 AMODE 31 AMODE ANY]</td> <td>[RMODE 24 RMODE ANY]</td> <td></td> </tr> <tr> <td>[370]</td> <td>[XA]</td> <td></td> </tr> </table>	[MAP NOMAP]	[STR NOSTR]	[OS DOS ALL]	[CLEAN NOCLEAN]	[SYSTEM]		[AMODE 24 AMODE 31 AMODE ANY]	[RMODE 24 RMODE ANY]		[370]	[XA]	
[MAP NOMAP]	[STR NOSTR]	[OS DOS ALL]											
[CLEAN NOCLEAN]	[SYSTEM]												
[AMODE 24 AMODE 31 AMODE ANY]	[RMODE 24 RMODE ANY]												
[370]	[XA]												

Operands

fn

is the file name of the MODULE file being created. If *fn* is not specified, the file created has a file name equal to that of the first entry point in the LOAD MAP.

fm

is the file mode of the MODULE file being created. If *fm* is not specified, A1 is assumed.

Options

If conflicting options are specified, the last one entered is used.

FROM *entry1*

specifies an entry point or a control section name that represents the starting virtual storage location from which a non-relocatable MODULE is generated.

TO *entry2*

specifies an entry point or a control section name that represents the ending virtual storage location from which a non-relocatable MODULE is generated.

MAP

copies system loader table entries for the generated module into a map record that is included in the MODULE file. The record can contain up to 3276 map entries. You can issue the MODMAP command to display the module map. MAP is the default.

Using this option in conjunction with the NOPRES option on the LOADMOD command deletes previously loaded non-OS programs.

NOMAP

specifies that a module map is not to be contained in the MODULE file.

Note: If a module is generated with the NOMAP option, that module cannot later be loaded and started with the CMS LOADMOD and START commands. When NOMAP is specified, the information produced is not sufficient for the START command to execute properly. However, a module generated with the NOMAP option can later be invoked as a command; that is, it can be invoked if its file name is entered.

NOSTR

does not cause anything to be deleted when used in conjunction with the NOPRES option on the LOADMOD command. NOSTR is the default.

STR

deletes previously loaded non-OS programs when used in conjunction with the NOPRES option on the LOADMOD command.

OS

indicates that the program may contain OS macros and, therefore, should be executed only when CMS/DOS is not active. OS is the default.

DOS

indicates that the program contains VSE macros; CMS/DOS must be active (that is, SET DOS ON must have been previously invoked) in order for this program to execute. (See Usage Note 2).

Note: This option is valid only in a System/370 mode virtual machine. The generated module does not work in a 370-XA mode virtual machine because DOS must be OFF in that environment.

ALL

indicates that the program:

- Contains CMS macros and must be capable of running regardless of whether CMS/DOS is active or not
- Contains no VSE or OS macros
- Preserves and resets the DOS flag in the CMS nucleus
- Does its own setting of the DOS flags

Note: The ALL option is primarily for use by CMS system programmers. CMS system routines are aware of which environment is active and will preserve and reset the DOS flag in the CMS nucleus.

CLEAN

indicates that the module is to be cleaned from storage at the end of its execution. CLEAN is the default for relocatable modules.

Note: The CLEAN option cannot be specified for non-relocatable modules.

NOCLEAN

indicates that the module is to remain in storage until end-of-command (Ready;). NOCLEAN can be specified for either relocatable or non-relocatable modules. NOCLEAN is the default for non-relocatable modules.

SYSTEM

indicates that when the MODULE file is subsequently loaded, it is to have a storage protect key of zero.

AMODE

specifies the addressing mode in which the program will be entered in an 370-XA mode virtual machine. In a System/370 mode virtual machine, you may specify AMODE, although only 24-bit addressing is available. This allows you to create XA capable module files on a System/370 mode virtual machine. The AMODE defined by this option is placed in the header record of the MODULE file being created.

This option overrides the AMODE determined by the LOAD process. If you specify RMODE, but do **not** specify AMODE, the AMODE for the MODULE is determined from the following default criteria:

- If you specify RMODE ANY, the AMODE specification defaults to AMODE 31.
- If you specify RMODE 24, the AMODE defaults to the AMODE of the entry point determined by the LOAD process.

If you specify neither AMODE nor RMODE, the AMODE for the module is determined by the LOAD process. The valid AMODE values and their meanings are:

- | | |
|----|--|
| 24 | This entry point is to receive control in 24-bit addressing mode. |
| 31 | This entry point is to receive control in 31-bit addressing mode when running in a 370-XA mode virtual machine. Since 31-bit addressing is unavailable in a System/370 mode virtual machine, the entry point will receive control in 24-bit addressing mode when running in a System/370 mode virtual machine. |

ANY This entry point is capable of operating in either 24-bit or 31-bit addressing mode. It will receive control in the addressing mode of its caller. If the module is executed by name from the command line or from an EXEC, it will receive control in:

- 31-bit addressing mode in an 370-XA mode virtual machine
- 24-bit addressing mode in a System/370 mode virtual machine.

RMODE

specifies, in an 370-XA mode virtual machine, the location in virtual machine storage where the loaded MODULE is to reside.

In a System/370 mode virtual machine, you may specify RMODE, although only 16Mb of storage is available. This allows you to create XA capable MODULE files on a System/370 mode virtual machine. The RMODE defined by this option overrides the RMODE determined by the LOAD process if the module being generated is relocatable. This means that you specified the RLDSAVE option on:

- the LOAD command
- any INCLUDE command you used to prepare the TEXT needed for the MODULE file.

The RMODE option will not provide override capability as stated if the module being generated is non-relocatable. This means that you did not specify the RLDSAVE option on the LOAD or INCLUDE command used for the MODULE file.

When you load a non-relocatable MODULE file using one of the CMS loading methods (except NUCXLOAD), the module will be loaded according to its generated origin.

Note: An AMODE value specified without an RMODE option defaults to RMODE 24 for the module.

If you specify neither AMODE nor RMODE, the RMODE for the module is determined by the LOAD process. The valid RMODE values and their meanings are:

24 The load must reside below the 16Mb line in a 370-XA mode virtual machine, overriding the RMODE determined by the LOAD process. A definition of RMODE 24 would be reflected in the MODULE header record. In a System/370 mode virtual machine, the load can only reside below 16Mb.

ANY The load may reside above or below the 16Mb line in a 370-XA mode virtual machine, overriding the RMODE determined by the LOAD process. A definition of RMODE ANY would be reflected in the MODULE header record. In a System/370 mode virtual machine, the load can only reside below 16Mb.

370

specifies that this module can execute in a System/370 mode virtual machine only.

XA

specifies that this module can execute in a 370-XA mode virtual machine only.

Notes:

1. If you specify neither 370 nor XA, no architecture attribute is added to the module. You can execute it in either a System/370 or 370-XA mode virtual machine. This is the default and is the same as specifying both 370 and XA.
2. If you specify both architectures, you can execute the module in either virtual machine architecture.

Usage Notes

1. The GENMOD command is usually invoked following the LOAD command, and possibly the INCLUDE command. For example, the sequence:


```
load myprog
genmod testprog
```

 loads the file MYPROG TEXT into virtual storage and creates a non-relocatable load module named TESTPROG MODULE. TESTPROG may now be invoked as a user-written command from the CMS environment.
2. The execution of MODULE files created from VSE programs is not supported and may give unpredictable results. GENMOD is intended for use with the LOAD command, not the FETCH command. Storage initialization for FETCH is different from that for LOAD.
3. Before the file is written, undefined symbols are set to location zero and the common reference control section is initialized. The undefined symbols are not retained as unresolved symbols in the MODULE file. Therefore, once the MODULE file is generated, those references cannot be resolved and may cause unpredictable results during execution.
4. If you load a program into the transient area, you should be careful not to exceed two pages (8192 bytes).
5. A transient module (loaded with the ORIGIN TRANS option) that was generated with the SYSTEM option is written on a disk or directory as a fixed-length record with a maximum length of 8192 bytes. The relocation and history information for these files cannot be saved.
6. If you are using FORTRAN under CMS, compiling with the RENT option, and have named the main program the same as *fn*, you must use the FROM option specifying *entry1* the same as *fn*, preceded by the "at" sign (@). For example, you would enter:


```
genmod mnprog (from @mnprog
```
7. If FROM is not specified on the GENMOD command, the starting virtual storage location (entry point) of the module is either the address of *fn* (if it is an external name) or the entry point determined according to the hierarchy discussed in Usage Note 3 on page 316 of the LOAD command. This is not necessarily the lowest address loaded. If you have any external references before your START or CSECT instructions, you must specify the 'FROM *entry1*' operand on the GENMOD command to load your program properly.
8. If you are using PL/I under CMS, use FROM PLISTART as an option to avoid unpredictable results.
9. You can use the GENMOD command to create a relocatable CMS MODULE file. Relocation is performed when using the LOADMOD or NUCXLOAD command. When the GENMOD command generates a MODULE file, one

record of this MODULE file can contain relocation information. This depends on the previous LOAD or INCLUDE command that was issued.

10. If the RLDSAVE option was specified on any of the LOAD or INCLUDE commands used to create this MODULE file, the file is considered to be relocatable. If the RLDSAVE option was not specified on any of the LOAD or INCLUDE commands, the relocation information for the file cannot be saved. See the LOAD or INCLUDE commands for more information.
11. The FROM or TO options should not be specified when module relocation information is being saved (using RLDSAVE option of the LOAD and INCLUDE commands). If specified, the results will be unpredictable.
12. If the HIST option was specified on the LOAD or INCLUDE command, the MODULE file may contain history information. See the LOAD or INCLUDE commands for more information.

Example

Example of a Non-relocatable Module:

The TEXTs specified by the LOAD and INCLUDE commands resulted in the load residing below 16Mb. Although you did not specify RLDSAVE, the RLD data associated with the TEXT files was processed to resolve addresses to their current storage locations.

Therefore, when you create a MODULE file with the GENMOD command:

- the RLD data associated with the TEXT files is not propagated to the MODULE file.
- the relocatable addresses in the executable code are updated to reflect the storage location where the MODULE was loaded.

When the MODULE file is loaded, it must reside in the storage locations used when it was created. This ensures that the executable code functions properly.

Messages and Return Codes

DMSERD257T	Internal system error at address <i>addr</i> (offset <i>offset</i>)
DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC = 31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSMOD003E	Invalid option: <i>option</i> [RC = 24]
DMSMOD005E	No <i>option</i> specified [RC = 24]
DMSMOD018E	No load map available [RC = 40]
DMSMOD021E	Entry point <i>name</i> not found [RC = 40]
DMSMOD032E	Invalid filetype <i>ft</i> [RC = 24]
DMSMOD037E	Filemode <i>mode</i> [(<i>vdev</i>)] is accessed as read/only [RC = 36]
DMSMOD040E	No files loaded [RC = 40]
DMSMOD069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSMOD070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSMOD084E	The length of the module to be generated is non-positive. GENMOD terminated. [RC = 24]
DMSMOD105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 31 55 70 76 99 100]
DMSMOD810E	370 cannot be specified as an architecture when AMODE is 31. [RC = 68]
DMSMOD811E	[AMODE 24 RMODE 24] cannot be specified when module size exceeds 16Mb. <i>file</i> not generated. [RC = 68]
DMSMOD943E	Invalid AMODE <i>mode</i> specified. <i>file</i> not generated. [RC = 24]

GENMOD

DMSMOD944E Invalid RMODE *mode* specified. *file* not generated. [RC = 24]
DMSMOD945E AMODE/RMODE values conflict. *file* not *generated*|*loaded*.
[RC = 68]
DMSMOD1298E CLEAN cannot be specified for a non-relocatable module. *fn* not
generated. [RC = 68]

Additional system messages may be issued by this command. The reasons for these
messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

GENMSG

Use the GENMSG command to convert a message repository file into an internal form. Each record is read from the input file, the syntax is checked, and it is placed in an output file in a form the message processor can use.

Format

GENMSG	<p><i>fn ft fm applid</i> [<i>langid</i>] [(options ...)]</p> <p>Options:</p> <p>[CP] [<u>Dbcs</u> NODbcs] [<u>List</u> NOList] [<u>Xref</u> NOXref]</p> <p>[<u>Object</u> NOObject] [<u>Margin nn</u> <u>Margin 72</u>]</p>
---------------	---

Operands

fn ft fm

is the file ID of the external repository to be converted. This file must be in fixed-record format, and have a LRECL of 80.

applid

specifies the application for which this repository is intended. This application identifier is 3 characters long.

langid

specifies the language whose repository should be used. The *langid* is 1 to 5 characters long. The default value is the language currently in use.

Options

CP

notes that the input file contains messages for CP. The output repository file is modified accordingly.

Dbcs

specifies that the input file contains Double-Byte Character Set (DBCS) characters as part of its message text.

NODbcs

specifies that the input file does not contain Double-Byte Character Set (DBCS) characters. NODBCS is the default.

List

creates a listing file from the compilation. This file has the same file name as the input file, and a file type of LISTING. LIST is the default.

NOList

does not create a listing file from the compilation.

Xref

creates a cross-reference of message text and offset within the repository and place it in the listing file.

NOXref

does not create a cross-reference. NOXREF is the default.

Object

creates an object file (a repository in internal form). This file has the same file name as the input file, but the file type will be TEXT. OBJECT is the default.

NOObject

does not create an object file.

Margin *nn*

shows that message source is taken from columns 1-*nn* in the input deck. The default is MARGIN 72.

Usage Notes

1. To learn how to make your own message repository refer to the publication, *VM/SP Application Development Guide for CMS*.
2. If you wish to change a given CMS message, build your own message repository and include the number of the CMS message that you want to override. Load your repository as a user repository using the SET LANGUAGE command. See the example below.

Unpredictable results may occur if the source message repository for CMS (DMSMES REPOS) is altered, recompiled, and used to build CMS.

3. GENMSG creates two output files:

fn TEXT fm	An internal form of the repository (object file)
fn LISTING fm	The compiler listing

The file mode for these output files is the same as the input file's file mode, unless the input file is read/only; in that case, the files go to the user's first available disk or directory accessed read/write. If no disk or directory exists with read/write access, an error message is returned, and execution terminates.

4. The object file that GENMSG produces must be properly loaded into storage for it to be included with an application in a national language. Load the repository as a user repository and use the SET LANGUAGE command to load the object file.
5. GENMSG displays an error message whenever it finds a syntax error in a message repository. Use the NOOBJECT option if you just want to check for syntax errors in your repository.
6. If your message repository will be updated, or if you are using IBM-supplied message repositories, then specify the MARGIN *nn* option as MARGIN 63.

Example

Assume you are working in an English language application called MYAPPL1 (*applid* = MYA). You already have created a small message repository for MYAPPL1, and it has a file ID MYOWN MESSAGES A. This repository contains a message with two formats and a message with five formats. To compile your repository, you enter:

```
genmsg myown messages a mya (margin 63
```

The following LISTING file is created by the message compiler:

```
GENMSG Version 1 Release 1.0 Page 1 Time 17:48:50 Date 85.248
```

```
Options used: MARGIN 63
Substitution character is &
Number of message number characters to display is 3
```

```
GENMSG Version 1 Release 1.0 Page 2 Time 17:48:50 Date 85.248
```

```
MESSAGE ID
NUM FMT LINE SEV TEXT
*
0001 01 01 E No filename specified 00002000
0001 02 01 E No &1 names specified 00003000
0002 01 01 E File &1 &2 &3 not found 00004000
0002 02 01 E &1 file &2 not found 00006000
0002 03 01 E Dataset not found 00007000
0002 04 01 E File(s) &1 not found 00008000
0002 05 01 E Note &1 not found 00009000
0002 05 01 E Note &1 not found 00010000
```

```
GENMSG Version 1 Release 1.0 Page 3 Time 17:48:50 Date 85.248
```

```
Total Messages Informational Warning Error Severe Terminating
2 2 0 0 0 0
```

```
The text deck is 000000F8 bytes in length
Return code was 0
```

Messages and Return Codes

```
DMSFNS1144E Implicit rollback occurred for work unit workunitid [RC=31]
DMSFNS1252T Rollback unsuccessful for file pool filepoolid
DMSMGC002E File fn ft fm not found [RC=28]
DMSMGC003E Invalid option: option [RC=16]
DMSMGC006E No read/write filemode accessed [RC=36]
DMSMGC029E Invalid parameter parameter in the option option field [RC=24]
DMSMGC049E Invalid line number nn [RC=8]
DMSMGC054E Incomplete fileid specified [RC=24]
DMSMGC065E option option specified twice [RC=24]
DMSMGC066E option1 and option2 are conflicting options [RC=24]
DMSMGC104S Error nn reading file fn ft fm from disk or directory [RC=100]
DMSMGC105S Error nn writing file fn ft fm to disk or directory [RC=100]
DMSMGC109S Virtual storage capacity exceeded [RC=104]
DMSMGC147E Message not in ascending sequence [RC=8]
DMSMGC580E Invalid string : unmatched shift-out (SO) and shift-in (SI) [RC=5]
DMSMGC580E Invalid string : invalid double-byte character(s) [RC=5]
DMSMGC766I Substitution character is char
DMSMGC767I Number of message number characters to display is nn
```

- DMSMGC768W Invalid substitution character value *char* [RC=4]
- DMSMGC769W Invalid number of message characters value *value* [RC=4]
- DMSMGC770E Invalid application ID *applid* [RC=16]
- DMSMGC771E Invalid message number- [RC=8]
- DMSMGC772E Invalid format number [RC=8]
- DMSMGC773E Duplicate message ID *id* [RC=4]
- DMSMGC774E Line numbers for messages are not consecutive [RC=8]
- DMSMGC775W Text too long - 240 characters is the maximum allowed [RC=4]
- DMSMGC776I Options used: *list*
- DMSMGC1262S Error *nm* opening file *fn ft fm*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

GLOBAL

Use the GLOBAL command to identify which CMS, CMS/DOS, or OS libraries are to be searched for macros, copy files, subroutines, VSE executable phases, CSL routines, or modules when processing subsequent CMS commands. The libraries may reside on CMS disks or in SFS directories.

Format

GLobal	$\left. \begin{array}{l} \text{MACLIB} \\ \text{TXTLIB} \\ \text{DOSLIB} \\ \text{LOADLIB} \\ \text{CSLLIB} \end{array} \right\} [\text{libname1...libname63}]$
---------------	---

Operands**MACLIB**

precedes the specification of macro libraries that are to be searched for macros and copy files during the execution of language processor commands. The macro libraries may be CMS files or OS data sets. If you specify an OS data set, a FILEDEF command must be issued for the data set before you issue the GLOBAL command.

TXTLIB

precedes the specification of text libraries to be searched for missing subroutines when the LOAD or INCLUDE command is issued, or when a dynamic load occurs (that is, when an OS SVC 8 is issued).

Note: Subroutines that are called by dynamic load should (1) contain only VCONs that are resolved within the same text library member or (2) be resident in storage throughout the processing of the original CMS LOAD or INCLUDE command. Otherwise, the entry point is unpredictable.

DOSLIB

precedes the specification of DOS simulated core image libraries (that is, CMS/DOS phase libraries) to be searched for missing phases. This operand does not apply to system or private core image libraries residing on DOS disks. DOSLIB can be specified regardless of whether the CMS/DOS environment is active or not.

LOADLIB

precedes the specification of load module libraries to be searched for a module that the OSRUN command or the LINK, LOAD, ATTACH, or XCTL macros refer to. The libraries can be CMS LOADLIBS or OS module libraries. If you specify an OS data set, issue a FILEDEF command for the data set before you issue the GLOBAL command.

CSLLIB

precedes the names of callable services libraries (CSLs). The GLOBAL CSLLIB command manipulates the library search order that is used by the RTNLOAD command to locate CSL routines. Saved segments are searched first for these libraries, then accessed disks and directories.

GLOBAL

Note: VMLIB is the name of the callable services library supplied with VM/SP. VMLIB is always implicitly in the search order— if the CSLLIB operand is specified without any library names, VMLIB is still available. If VMLIB is specified, it is searched in the order given; if it is not specified, it is searched last.

libname1...

are the file names of up to 63 libraries of the specified file type (MACLIB, TXTLIB, DOSLIB, LOADLIB, or CSLLIB). The libraries are searched in the order in which they are named. The library list is subject to other system limits, such as command line length. This command supersedes any previous GLOBAL command for the specified file type. If no file names are specified, the command cancels any previous GLOBAL command for this file type.

Usage Notes

1. A GLOBAL command remains in effect for an entire CMS session unless it is explicitly canceled or reissued. If a program failure forces you to IPL CMS again, you must reissue the GLOBAL command.
2. There are no default libraries; if you wish to use the same libraries during every terminal session, place the GLOBAL command(s) in your PROFILE EXEC.
3. If you want to use an OS library during the execution of a language processor, you can issue a GLOBAL command to access the library, as long as you have defined the library via the FILEDEF command. If you want to use that library for more than one job, however, you should use the PERM option on the FILEDEF command, since the language processors clear nonpermanent file definitions.
4. To find out what libraries have been specified, issue the QUERY command with the MACLIB, TXTLIB, DOSLIB, LOADLIB, CSLLIB, or LIBRARY operands. (The LIBRARY operand requests a display of all libraries.)
5. For information on creating and/or manipulating CMS libraries, see the discussion of the MACLIB, TXTLIB, DOSLIB, LOADLIB, and CSLGEN commands. (See the *Application Development Guide for CMS* for a description of the CSLGEN command.)

Example

To specify that you want the CMSLIB, OSMACRO, and OSMACRO1 macro libraries searched when processing CMS commands, you would enter:

```
global maclib cmslib osmacro osmacro1
```

If you enter

```
global csllib userlib1 userlib2
```

the CSL search order becomes: USERLIB1 USERLIB2 VMLIB, with VMLIB implicitly included at the end of the search order. Issuing a

```
query csllib
```

will display

```
CSLLIB = USERLIB1 USERLIB2
```

Messages and Return Codes

DMSERD257T	Internal system error at address <i>addr</i> (offset <i>offset</i>)
DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC = 31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSGLEB002W	File <i>fn ft [fm]</i> not found [RC = 4 or 28]
DMSGLEB014E	Invalid function <i>function</i> [RC = 24]
DMSGLEB047E	No function specified [RC = 24]
DMSGLEB056E	File <i>fn ft [fm]</i> contains invalid record formats [RC = 32]
DMSGLEB104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 31 55 70 76 99 100]
DMSGLEB108S	More than <i>nn</i> libraries specified [RC = 88]
DMSGLEB109S	Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

GLOBALV

The GLOBALV (GLOBAL Variables) command addresses two primary needs: 1) the need for several execs to share a common set of values; 2) the need to retain those values, either temporarily or permanently, for subsequent use.

Sharing:

Values are often given names, describing what they represent, for easy reference. Although the values often vary, their names usually do not. The GLOBALV command processor builds and maintains group(s) of named, variable values in free storage for shared use by execs. Execs “share” a value by referring to it by a common name. When requested, GLOBALV retrieves a variable(s) from the group(s) and places it in the program stack for subsequent use by the requesting exec.

GLOBALV supports use of more than one group. This allows for grouping distinct variables that are either related or often used together, which facilitates both more efficient retrieval and more selective use. The “global variable group(s),” built by GLOBALV from a set of CMS GLOBALV type files on the user’s disk or directory accessed as A and extensions, exist throughout an IPL, unless explicitly purged or re-initialized.

Retaining:

When variables are defined or changed, the user decides whether the variables or changes are to last:

1. For the current IPL only
2. Throughout an entire session (normally, from LOGON to LOGOFF)
3. Permanently, i.e. across sessions

Variables defined for the current IPL only are retained in storage. Those required longer than a single IPL are retained in CMS files on the user’s disk or directory accessed as A from where they are put in storage. The CMS file names are SESSION GLOBALV (for values required throughout the session), and LASTING GLOBALV (for values that are to last permanently). These two files and a third file (INITIAL GLOBALV) are the source from which the GLOBALV command processor creates and initializes the variable(s) in storage. The INITIAL file is normally created by the user as an alternative way of defining a large number of variables for an IPL.

The CMS GLOBALV files may be of fixed or variable format. Fixed format facilitates creation of files by users (via editing). It accommodates variables whose names and values do not exceed eight bytes each. The GLOBALV command processor uses variable format which allows for varying length variable names and values. In addition, variable format includes a special field which, when used, identifies the group name into which the variable will be grouped. See Usage Note 1 below for additional information about fixed and variable formats.

The GLOBALV command processor manages requests to define or set (SET...) variables both in storage and in the LASTING and SESSION GLOBALV files on the user’s disk or directory accessed as A.

Format

GLOBALV	<pre> INIT SELECT [group UNNAMED] [SELECT {group UNNAMED} SET SETS {name1 [value1] [name2 value2] }.. SETP SETL SETLS {name [value] } SETSL SETLP SETPL LIST [name1 [name2] ...] STACK {name1 [name2] ...} STACKR SELECT {group UNNAMED} PUT PUTS {name1 [name2] ...} PUTP GET [name1 [name2] ...] SELECT {group UNNAMED} PURGE GRPLIST GRPSTACK PURGE </pre>
----------------	---

Note: Although this command (except for the GET and PUT options) may be used in CMS execs, it is designed for use with System Product Interpreter or EXEC 2 execs. Restrictions and precautions are listed in "Usage Notes for CMS Execs."

Operands

INIT

allocates and initializes global variable(s) in free storage from data in the LASTING, SESSION, and INITIAL GLOBALV files on the issuer's minidisk or directory accessed as A and extensions. If your LASTING GLOBALV file resides in an SFS directory and using INIT rewrites your LASTING file causing it to be empty, then the file does not get erased as with minidisks. GLOBALV automatically inserts two dummy records with group name DMSGLO into the empty file. This preserves any SFS authorities on the file. When rewriting of the LASTING file does not result in an empty file, the dummy records are removed.

Not all files need to be present. GLOBALV performs any needed cleanup (to eliminate multiple and null entries) in the LASTING GLOBALV file.

If the records in the GLOBALV file(s) contain no group name, for grouping the variables, (as with fixed format records) GLOBALV's INIT function allocates only one global variable group, UNNAMED, in free storage. Otherwise,

(variable format) GLOBALV INIT will allocate as many unique global variable groups in free storage as identified in the GLOBALV files.

GLOBALV INIT initializes free storage with variables defined in the LASTING, SESSION, and INITIAL GLOBALV files respectively. If any variables are defined more than once within the LASTING file or within the SESSION file, the value defined last in the file is the one used to initialize storage. If a variable of the same name is defined in both the LASTING and SESSION files, the value assigned in the SESSION file will *override* the value assigned in the LASTING file when the storage is initialized. (See "Usage Notes 2 and 3.")

After initializing free-storage from the LASTING GLOBALV file, the file is re-written to eliminate multiple definitions for any variable names and any null (zero length) value assignments.

The global variable(s) in free storage are required by all other GLOBALV functions. Therefore, GLOBALV INIT is performed automatically if not explicitly requested prior to other GLOBALV requests.

SELECT

identifies the global variable group which is the subject of this or subsequent SELECT sub-functions. If no sub-function is specified, the GLOBALV command processor interprets the command as a request to set the default group for subsequent SELECT sub-functions. The default is set to the group indicated by "group" or to UNNAMED if no group is specified. A GLOBALV SELECT command that *does* specify a sub-function affects only the group specified in the command. It has no effect on setting or resetting the default group.

The SELECT phrase (SELECT group or SELECT UNNAMED) is optional preceding all forms of the SELECT sub-functions, SET, PUT, GET, LIST, and STACK. If the SELECT phrase is not used, the sub-function affects the default group, described above. (See "Examples" for uses of GLOBALV SELECT.)

SELECT Sub-functions:**SET****SETS****SETP**

assigns the value "value1" to the variable "name1," the value "value2" to the variable "name2," etc. Since SET fields are delimited by blanks, the values cannot contain any blanks. (Use the SETL sub-function, described next, for such values.) If no value is specified, the value is assumed to be null.

SET adds the assignment(s) in the selected or default global variable group in storage. If the variable name is already defined, its value is replaced by the one specified in the command. SETS adds/replaces the assignment(s) in the selected or default group and appends it to the SESSION GLOBALV disk file. SETP adds/replaces the assignment(s) in the selected or default group and appends it to the LASTING GLOBALV file.

CMS exec users, refer to "Usage Notes for CMS execs."

SETL**SETLS****SETSL****SETLP****SETPL**

assigns the specified literal value, which may contain blanks, to the variable name. The first blank following the name delimits the name from the value field and is not part of the value. All characters following this blank (including any

other blanks) are part of the value. If no value is specified, the value is assumed to be null.

SETL adds the assignment in the selected or default global variable group in storage. If the variable name is already defined, its value replaced by the one specified in the command. SETLS adds/replaces the assignment in the selected or default group and appends it to the SESSION GLOBALV file. SETSL is equivalent to SETLS. SETLP adds/replaces the assignment in the selected or default group and appends it to the LASTING GLOBALV file. SETPL is equivalent to SETLP.

CMS exec users, refer to “Usage Notes for CMS Execs.”

LIST

displays a list of the specified variable name(s) and their associated value(s). If no name is specified, all variables in the selected or default group are listed.

STACK

places the value(s) associated with the specified variable name(s), from the selected or default group, LIFO in the program stack. When multiple variables are named in a single stack request, the values are stacked LIFO in the program stack such that the variable named first in the command is the first retrieved from the stack. Refer to Example 2 under “EXAMPLES.” If a variable is not found in the group, a null (zero length) line is stacked. The command has no effect if the variable name is omitted.

This stacking technique requires that the System Product Interpreter exec issue a separate “PULL” or PARSE PULL control statement to read each value from the stack.

STACKR

places a “&READ n” control statement and, for each variable name specified, a “&name = &LITERAL OF value” assignment statement LIFO in the program stack such that “&READ n” is the first retrievable line. In the &READ control statement, “n” is the number of subsequent assignment statements and, in the assignment statement, “value” is the value associated with the specified variable name in the selected or default group. When multiple variables are named in a single STACKR request, the values are stacked LIFO in the program stack such that the “&READ n” is the first retrievable line from the stack, and the first named variable assignment statement is the next retrievable line from the stack, etc. Refer to Example 1 in the “Examples” section. The command has no effect if the variable name is omitted.

This stacking technique requires only a single &READ control statement to read all the variables named on the GLOBALV command from the stack. The STACKR option only applies to EXEC and EXEC 2 execs.

CMS exec users, refer to “Usage Notes for CMS Execs.”

PUT

PUTS

PUTP

determines the value of the EXEC 2 or System Product Interpreter variable(s) specified in ‘name1’, ‘name2’, etc. and assigns that value as a global value in the selected or default global variable group. If a global value already exists with the name specified, it is replaced with the specified name’s value.

PUT adds or replaces the assignment(s) in the selected or default global variable group in storage. PUTS does the same, but appends the group to the SESSION

GLOBALV

GLOBALV file. PUTP does the same, but appends the group to the LASTING GLOBALV file.

Note: The PUT and GET sub-functions can only be used from an EXEC 2 or a System Product Interpreter exec, and are subject to the rules of the interpreter being used. Refer to the usage notes for additional details.

GET

assigns values from the specified or default global variable group to the specified EXEC 2 or System Product Interpreter variable names. If no names are specified, GET does nothing.

Note: The PUT and GET sub-functions can only be used from an EXEC 2 or a System Product Interpreter exec, and are subject to the rules of the interpreter being used. Refer to the usage notes for additional details.

PURGE

causes the variables in storage to be cleared. The group itself is not purged.

CAUTION:

If the SELECT phrase is not included with the PURGE sub-function the result will be a GLOBALV PURGE of all global variable(s) in storage.

GRPLIST

displays a list of all¹⁰ existing global variable groups.

GRPSTACK

places the names of all¹⁰ existing variable groups, line by line, in the program stack such that these items will be the first retrievable from the stack. A null line, used as a delimiter, indicates the end of the stacked group names.

PURGE

causes all¹⁰ global variable(s) in free storage to be released.

Usage Notes

1. The CMS GLOBALV files may be of fixed or variable format. Fixed format records are 16 bytes in length and consist of two eight-byte fields that contain a variable name, followed by its assigned value. Variable format records may be up to 520 bytes in length and consist of the following five fields (Because f1 and f2 can not be more than X'FF' or 255, any variable name or variable value that exceeds this maximum length will be truncated to its first 255 bytes.):

group name	f1	variable name	f2	variable value
0	8	9	n	n+1

group name

identifies the group for grouping the variable (from GLOBALV [SELECT group|UNNAMED] SET...).

f1

defines the actual length of the variable name field immediately following.

¹⁰ Note that "all" includes those groups created by use of the DEFAULTS and EXECUTE commands.

variable name

identifies the name by which this shared value will be commonly referenced.

f2

defines the actual length of the variable value field immediately following.

variable value

specifies the actual value assigned to the named variable.

Use fixed format when editing (creating or updating) files. Variable format records would be difficult to edit because changes in the variable name or variable value fields must also be reflected in their respective length fields. Although not impossible, this further editing is awkward and likely to be overlooked, increasing the chance of errors in those fields.

To establish the initial set of lasting variables, the user may edit them into a fixed format LASTING file. Note that whenever the GLOBALV command processor rewrites this file, during initialization, it will use variable format.

Probably the easiest way to create GLOBALV file(s) is to let the GLOBALV command processor do the work. Create an EXEC file containing the appropriate GLOBALV ... SETS and/or SETL commands. Then when the exec is invoked, the GLOBALV command processor will build the file(s) as it executes the commands.

2. The SESSION file is not erased by the GLOBALV command processor. This is the responsibility of the user. The length of a session is thus determined by the frequency with which the user erases the SESSION GLOBALV file. To make the duration of a session the time between CMS IPLs, the user might choose to include an ERASE SESSION GLOBALV command in the PROFILE EXEC. To make a session last for all IPLs of CMS during one day, erase the SESSION GLOBALV file whenever the date changes.

The SESSION GLOBALV file also is *never* cleaned up (to eliminate multiple and null entries) by the GLOBALV command processor, as the LASTING GLOBALV file is at each initialization. Without this automatic cleanup, the SESSION GLOBALV file continues to grow longer with each SETS and SETSL command.

3. If the file is present during initialization of the global variable(s) in storage, its variables take precedence over LASTING variables of the same name. For variables of the same name defined within a file or in more than one file, the order of precedence, is:

```
SESSION - last in file is used
LASTING - last in file is used
INITIAL - first in file is used
```

So, for example, if a variable were defined for a given group several times in each file, and all files were present at initialization, the value used in the storage would be that defined last in the SESSION GLOBALV file.

4. The GLOBALV function SETL does not preserve lower-case argument values when called from the following:
 - from the command line.
 - from an XEDIT macro without an &COMMAND preceding it.
 - from a System Product Interpreter exec with the ADDRESS CMS command.

5. The PUT and GET sub-functions use the EXECCOM facility of the interpreter currently executing, EXEC 2 or the System Product Interpreter (REXX), to set and retrieve variables.

For EXEC 2, names passed through EXECCOMM are taken exactly as specified, and embedded ampersands (&) do not cause multiple substitution.

For REXX, no substitution or case translation takes place. Simple symbols must be valid REXX variable names, that is, in uppercase, and not starting with a number or a period. However, in compound symbols, any characters (including lowercase, blanks, etc.) are permitted following a valid REXX stem.

Usage Notes for CMS Execs

1. When defining values using GLOBALV's SELECT sub-function, SET..., be aware that values (tokens) larger than eight characters will be truncated to eight characters by the CMS Exec processor.
2. Avoid use of GLOBALV's SELECT sub-function, SETL... . It requires an extended parameter list, such as is provided by EXEC 2. Use in CMS execs causes an error from the GLOBALV command processor.
3. Avoid use of GLOBALV's SELECT sub-function, STACKR. The literal assignment statement it generates is not in a format the CMS Exec processor recognizes. The CMS Exec command processor will generate the following error message:

```
DMSEXT072E      ERROR IN EXEC FILE 'fn' LINE 'nnn' - INVALID ASSIGNMENT
```

Examples

These examples illustrate the use and effect of several, consecutive GLOBALV SELECT commands.

Example 1:

<p>GLOBALV SET DEPT 222 (SELECT phrase is omitted.)</p> <p>Effect: The value "222" is assigned to variable name "DEPT" in the default global variable group "UNNAMED".</p>	<p>UNNAMED Group</p> <p>.</p> <p>.</p> <p>DEPT 222</p>
<p>GLOBALV SELECT TABA</p> <p>Effect: The default global variable group for subsequent SELECT sub-functions is set to "TABA"</p>	
<p>GLOBALV SET EMP 8888 MONTH MAY</p> <p>Effect: The value "8888" is assigned to the variable name "EMP" and the value "MAY" is assigned to the variable name "MONTH" in the default group "TABA".</p>	<p>TABA Group</p> <p>.</p> <p>.</p> <p>EMP 8888 MONTH MAY</p>
<p>GLOBALV SELECT UNNAMED SET YEAR 1982</p> <p>Effect: The value "1982" is assigned to the variable name "YEAR" in the global variable group "UNNAMED". The default setting is not changed.</p>	<p>UNNAMED Group</p> <p>.</p> <p>.</p> <p>DEPT 222 YEAR 1982</p>
<p>GLOBALV SETS YEAR 1981</p> <p>Effect: The value "1981" is assigned to the variable name "YEAR" in the default global variable group "TABA" and the assignment is entered into the SESSION GLOBALV file.</p>	<p>TABA SESSION Group GLOBALV File</p> <p>.</p> <p>.</p> <p>EMP 8888 YEAR 1981 MONTH MAY</p> <p>YEAR 1981</p>

GLOBALV STACK YEAR DEPT	Stack <i>Before After</i>
Effect: Places the value associated with variable name "YEAR" from group "TABA" onto the stack. Since the variable "DEPT" is not defined in global variable group "TABA", a null line is stacked.	Next line to read: ABC 1981 XYZ (null line) ABC XYZ

GLOBALV SELECT UNNAMED STACKR YEAR DEPT	Stack <i>After</i>
Effect: Places a "&READ 002" control statement, and two literal assignment statements, defining the variable name "YEAR" and the variable name "DEPT" with their associated values from global variable group "UNNAMED", onto the stack.	Next line to read: &READ 002 &YEAR = LITERAL OF 1982 &DEPT = LITERAL OF 222 1981 (null line) ABC XYZ

Example 2:

The effect of the following request, which names 3 variables:

GLOBALV SELECT TABA STACK EMP MONTH YEAR	Stack <i>After</i>
	Next line to read: 8888 MAY 1981

Whereas, the effect of 3 consecutive STACK requests, naming a single variable each (the same 3 variables as the multiple request above):

GLOBALV SELECT TABA GLOBALV STACK EMP GLOBALV STACK MONTH GLOBALV STACK YEAR	Stack <i>After</i>
	Next line to read: 1981 MAY 8888

Responses

GLOBALV ... LIST results in a display of the requested list.

GLOBALV GRPLIST results in a display of the requested list.

Messages and Return Codes

DMSERD257T Internal system error at address *addr* (offset *offset*)
 DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
 DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
 DMSGLO047E No function specified [RC=24]
 DMSGLO104S Error *nn* reading file *fn* GLOBALV A from disk or directory
 [RC=*nn*000]
 DMSGLO109S Virtual storage capacity exceeded [RC=400041]
 DMSGLO618E NUCEXT failed [RC=*nn*]
 DMSGLO622E Insufficient free storage; no table made [RC=*rc*]
 DMSGLO628E Invalid GLOBALV function *function* [RC=4]
 DMSGLO631E *function* can only be executed from an EXEC-2 or REXX EXEC
 [or as a CMS command] [RC=*rc*]
 DMSGLO649E Extraneous parameter *parameter* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814
Errors in copying a file	70
Errors in erasing a file	145

Error Codes

GLOBALV error codes consist of two 3-character fields. The first field corresponds to errors encountered during the GLOBALV INIT function; the second corresponds to errors encountered during other GLOBALV functions.

Code	Meaning
<i>nnn nnn</i>	
000 ...	Function completed successfully.
001	Truncation to the maximum length of 255 bytes occurred for variable name and/or variable value.
004	Invalid function/sub-function; or invalid environment for use of function/sub-function.
008	Error return from ATTN. Stacking suspended.
012	No free storage available to define (SET..) additional variables. Processing suspended at point of error.
024	No function specified on GLOBALV command.
1nn	I/O error appending newly defined variable(s) to LASTING or SESSION GLOBALV file on the user's disk or directory accessed as A. The assignment was, however, added to the appropriate global variable group in storage. Refer to FSWRITE macro for meaning of "nn."
5nn	EXECCOMM failed with return code "nn." For the meaning of "nn," refer to the EXECCOMM facility in the <i>VM/SP System Product Interpreter Reference</i> or the <i>VM/SP EXEC 2 Reference</i> , depending on the EXEC environment in which the failure occurred.

GLOBALV

-5nn	EXECCOMM failed with return code “-nn.” For the meaning of “-nn,” refer to the EXECCOMM facility in the <i>VM/SP System Product Interpreter Reference</i> or the <i>VM/SP EXEC 2 Reference</i> , depending on the EXEC environment in which the failure occurred.
1nn 000	I/O error reading GLOBALV type files from user’s disk or directory accessed as A. No global variable(s) in storage created. Refer to FSREAD macro for meaning of “nn.”
2nn ...	I/O error rewriting LASTING GLOBALV file into a temporary file. Global variables in storage are created, but rewrite of the LASTING file was suspended. The original LASTING file remains intact on the user’s disk or directory accessed as A. Refer to FSWRITE macro for meaning of “nn.”
000	Function completed successfully.
1nn	I/O error appending newly defined variable(s) to LASTING or SESSION GLOBALV file on the user’s disk or directory accessed as A. The assignment was, however, added to the appropriate global variable in storage. Refer to FSWRITE macro for meaning of “nn.”
3nn ...	Error occurred renaming the temporary LASTING GLOBALV file to become the new LASTING file. Global variables in storage are created. The original LASTING file was destroyed, but TEMPFILE GLOBALV contains its corresponding variables. Refer to RENAME command for meaning of “nn.”
000	Function completed successfully.
008	Error return from ATTN. Stacking suspended.
400 041	Error occurred when GLOBALV attempted to allocate storage for a workarea. GLOBALV initialization functions could not proceed.

GRANT AUTHORITY

Use the GRANT AUTHORITY command to authorize others to read from or write to one or more of your files or directories in the Shared File System. Use the QUERY AUTHORITY command to display authorities.

Format

GRAnt AUTHority	$\left[\begin{array}{cc} fn & ft \\ * & * \end{array} \right] \text{ dirid TO } \left\{ \begin{array}{l} \text{userid} \\ \text{nickname} \\ \text{PUBLIC} \end{array} \right\} [(\text{options...} [])]$ <p>Options: $\left[\begin{array}{l} \text{REAd} \\ \text{WRItE} \end{array} \right]$ $\left[\begin{array}{l} \text{TYPe} \\ \text{NOType} \\ \text{STACK} \left[\begin{array}{l} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{LIFO} \\ \text{FIFO} \end{array} \right]$</p>
------------------------	--

Operands

fn ft

identifies the file for which you are granting authority. Special characters (* and %) can be used to designate a set of files. See "Pattern Matching" on page 8 for information on these special characters. Subdirectories, revoked and erased aliases, and files that you do not own are ignored when using special characters.

dirid

identifies the name of the directory. If *fn ft* is specified, *dirid* identifies the directory in which the file is located. If *fn ft* are not specified, *dirid* identifies the directory for which authority is to be granted. For a detailed description of *dirid* see "Naming Shared File System (SFS) Directories" on page 4.

TO *userid*

TO *nickname*

specifies the name or set of names to whom you are giving read or write authority. Nicknames can refer to a group of users (see the NAMES command).

TO PUBLIC

indicates all users that can connect to the file pool are given authority.

Options

REAd

gives the user read authority to a file or directory. This is the default.

Read authority on a file allows the user to read the contents of a file.

Read authority on a directory allows the user to read the directory contents, which consists of the names of the files and subdirectories, if any. It also allows the user to access the directory (see ACCESS command).

GRANT AUTHORITY

WRite

gives the user all the privileges of read authority, plus:

- for a file, the authority to modify or erase the file. To erase or rename a base file, you must have write authority to the file and to the directory where it resides. To erase or rename an alias, you must have write authority for the alias, and read authority on the base file for that alias.
- for a directory, the authority to create files and aliases in that directory. It does not give authorization to read or write any of the files in that directory.

TYPe

displays the affected file(s) or directory at the terminal.

NOType

suppresses the display of affected file(s) or directory. **NOType** is the default.

STACK [FIFO]

STACK LIFO

places the information in the console stack rather than displaying it at the terminal. **FIFO** is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to **STACK FIFO**.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to **STACK LIFO**.

Usage Notes

1. You must own the file or directory to grant authority on it. For example, **USERA** creates a file and grants write authority to you. You cannot grant any authority on **USERA**'s file to any other user.
2. When a base file or directory is created, the owner always has write authority.
3. Read authority can be upgraded to write by reissuing the **GRANT AUTHORITY** command using the **WRITE** option.
4. To change an authority from write to read authority, you must issue a **REVOKE AUTHORITY** command with the **KEEPREAD** option.
5. When you grant authority to an alias, the authority refers to the base file. The alias is just a pointer to a base file. Similarly, when you have authority to a base file, you also have the same authority on any alias of the base file.
6. You can grant authority to a file or directory whether or not the file or directory is locked except when it is locked **EXCLUSIVE** by another user.
7. You can grant authority to a user ID even if the user ID is not enrolled in the file pool or does not exist in the VM system directory. This allows you to grant authority to a set of user IDs before they can get to the files or directories. These users will be able to use the files and directories after they are added to the VM system directory and are enrolled in the file pool.
8. You can grant authority to a file whether or not the file is open. This authority takes affect when the file is closed.
9. If special characters are specified to grant authority to a set of files, the directory must be closed.

10. If you wish to grant authority to a user ID of PUBLIC, or any of the abbreviations of the PUBLIC operand (PUB, PUBL, and PUBLI), then create a nickname for that user ID using the NAMES command. Use the nickname that you created when issuing the GRANT AUTHORITY command.

When you are specifying a nickname, notice the NODE tag in the NAMES file. If the NODE tag indicates that the user is on another processor, the LOCALID tag must also be specified. See the NAMES command for more information.

11. You cannot use the GRANT AUTHORITY to grant authority to a file or directory that is ESM (External Security Manager) protected. You must use an ESM command to do this.

Messages and Return Codes

DMSJAU002E	File <i>fn ft fm dirname</i> not found [RC=28]
DMSJAU065E	<i>option</i> option specified twice [RC=24]
DMSJAU066E	<i>option1</i> and <i>option2</i> are conflicting options [RC=24]
DMSJAU391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSJAU1163E	GRANT AUTHORITY failed for <i>fn ft dirname</i> [RC= <i>nn</i>]
DMSJAU1184E	File <i>fn ft fm dirname</i> not found or you are not authorized for it [RC=28]
DMSJAU1184E	Directory <i>dirname</i> not found or you are not authorized for it [RC=28]
DMSJAU1198E	Directory <i>dirname</i> is currently open; it must be closed before you can change the authority to any file in it [RC=70]
DMSJAU1210E	Directory <i>dirname</i> not found [RC=28]
DMSJAU1242E	External security in effect for <i>fn ft fm dirname</i> GRANT AUTHORITY command cannot be used [RC=88]
DMSJAU1243W	User <i>userid</i> already has WRITE authority to <i>fn ft fm dirname</i> [RC=4]
DMSJAU1243W	At least one user in the list (<i>userid</i>) already has WRITE authority to <i>fn ft fm dirname</i> [RC=4]
DMSJAU1246W	Public WRITE authority already granted on <i>fn ft fm dirname</i> [RC=4]
DMSJAU1287W	You do not own file <i>fn ft fm dirname</i> [RC=4]
DMSJAU1293I	You have granted authority to all users of the file pool. [RC=0 4]
DMSJED069E	Filemode <i>mode</i> not accessed [RC=36]
DMSJED109S	Virtual storage capacity exceeded [RC=104]
DMSJED1187E	Too many subdirectory levels in <i>dirname</i> [RC=24]
DMSJED1188E	Filemode <i>mode</i> is not associated with a directory [RC=74]
DMSJED1189E	Filemode <i>mode</i> is associated with a top directory [RC=24]
DMSJED1223E	There is no default file pool currently defined [RC=40]
DMSJNL637E	Missing nodeid for the AT operand [RC=24]
DMSJNL647E	Userid not specified for <i>nickname</i> in <i>userid</i> NAMES file [RC=32]
DMSJNL647E	Localid not specified for <i>userid</i> at <i>node</i> in <i>userid</i> NAMES file [RC=32]
DMSJNL653E	Error executing <i>command</i> rc= <i>nn</i> [RC=40]
DMSOUT065E	<i>option</i> option specified twice [RC=24]
DMSOUT066E	<i>option</i> and <i>option</i> are conflicting options [RC=24]
DMSOUT1201E	STACK option cannot follow FIFO or LIFO [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

GRANT AUTHORITY

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

HELP

Use the HELP command to display online HELP. The HELP facility can contain three layers of information:

1. BRIEF - concise description and example
2. DETAIL - complete information (including messages)
3. RELATED - information about similar commands.

HELP files are available for the following:

- Commands: AVS (APPC/VM VTAM support), CMS, CP, CPOTHER, GCS, SQL/DS (Structured Query Language Data System), and TSAF (Transparent Services Access Facility).
- Instructions: EXEC, EXEC2, and REXX (Restructured Extended Executor).
- Macros: CMS assembler language
- Routines: From VMLIB callable services library (CSL)
- Subcommands: EDIT and XEDIT, IPCS (Interactive Problem Control System), and SRPI (Server-Requester Programming Interface).

Format

Help	<pre> [<u>TASKs</u> Help taskname TASKs menuname MENU component-name cmd-name] [<u>MESSAGE</u>] message-id MSG <u>OptionA:</u> [<u>BRIef</u> <u>DETail</u> <u>RELated</u>] <u>OptionB:</u> [<u>ALL</u>] [<u>DESCript</u>] [<u>FORMat</u>] [<u>PARMs</u>] [<u>OPTions</u>] [<u>NOTEs</u>] [<u>ERRors</u>] <u>OptionC:</u> [<u>SCREen</u>] [<u>TYPE</u>] [<u>EXTend</u>] [<u>NOScreen</u>] [<u>NOType</u>] </pre>
-------------	---

Operands

Help, specified without any parameters, displays a task menu if you are using the VM/SP HELP files. If you are using files other than the VM/SP HELP files and if the HELP HELPMENU file has been created, then you will get the HELP HELPMENU file.

TASKs

displays the main TASK menu and is an easy way to begin using HELP. This menu leads to other task panels, short explanations for getting message information or for using the HELP facility, and the major MENU panels. TASK is the default if no parameters are specified in the HELP command and if a user-created file HELP HELPMENU does not exist.

Help

displays a BRIEF HELP for the HELP command with a command description, its format, an example, and, if applicable, a message indicating that additional help is available. To receive at your terminal the information you're now reading, enter

help help (detail)

taskname **TASKs**

displays the specified TASK file. TASK panel selections may include other TASK menus, a MENU panel, or a command HELP file. The TASK panels are organized in a branching structure of general to specific tasks. For example:

Taskname	Description
DISK	Disk operations
MANAGE	File management
INQUIRE	System information
QPRINTER	CP QUERY PRINTER command
TAPEDUMP	CMS TAPE DUMP command
WINDOW	Window manipulation
BORDER	Windowing border commands

The general TASKs files contain task menus that lead you to the specific task that you want information about. For example, entering:

help disk tasks

displays a task menu listing the specific tasks concerning disks. You can enter any of the specific TASKs file names (one that have a file type of HELPTASK) as well as the general ones.

In addition to the supplied tasknames, installations may create their own taskname files.

menuname **MENU**

displays a selection of HELP files. (They may be command or other menus.) The menuname is usually the name of a component. For example, if you want to display the menu of CMS commands, you would enter HELP CMS MENU.

component-name command-name

displays the HELP file for the specified command-name. The command-name can be the name of a command, a subcommand, or a statement. The component-name is the name of the component (or grouping) about which you want information. The HELP facility has the following major components:

Help Component Description

AVS	APPC/VM VTAM support commands
CMS	Conversational Monitor System commands, macros, messages, and routines.
CMSQUERY	CMS QUERY commands
CMSSET	CMS SET commands
CP	Control Program commands for general users

CPOTHER	Control Program privileged commands for other than general user
CPQUERY	CP QUERY commands
CPSET	CP SET commands
EDIT	EDIT subcommands
EXEC	EXEC statements
EXEC2	EXEC 2 statements
IPCS	Interactive Problem Control System subcommands, and includes Group Control System (GCS) subcommands
MACRO	CMS assembler language macros
PREFIX	XEDIT prefix commands
PVM	VM/Pass-Through Facility (Program Product 5748-RC1) if installed on your system.
ROUTINE	From the callable services library (CSL)
QUERY	XEDIT QUERY subcommands
RSCS	Remote Spooling Communications Subsystems Networking (Program Product 5664-188) if installed on your system.
SET	XEDIT SET subcommands
XEDIT	XEDIT subcommands
REXX	System Product Interpreter Statements
SQLDS	SQL/Data System Program Product (5748-XXJ) (only if you have this installed on your system.)
SRPI	Server-Requester Programming Interface subcommands
TSAF	Transparent Services Access Facility commands

In addition to these components, installations may create their own components.

If a component is not specified, then the search order for HELP files is:

1. CMS
2. CP
3. MENU
4. TASK
5. MSG

Therefore, if you specify

help command-name

HELP searches for the following:

1. CMS command-name
2. CP command-name
3. command-name MENU
4. command-name TASK
5. MSG command-name

MESSAGE *message-id*

MSG

displays the HELP file for a message. 'message-id' is the 7, 8, 10, or 11-character message-id you specify to display the HELP file for a message. Specify the message-id in one of four forms:

HELP

xxxxnnt
xxxxnnnt
xxxxmmnnnt
xxxxmmnnnt

where:

xxx

indicates the component (for example, DMS for CMS messages, DMK for CP messages).

mmm

is the module identifier.

nnn or nnnn

is the message number.

t

is the message type.

For example, for information on a message, you can specify either the 7- or 8-character message-id:

DMS250S
DMS0250S

or the 10- or 11-character message-id that also identifies the issuing module:

DMSHLP250S
DMSHLP0250S

Options

HELP files may contain different layers of information. You can get to these different layers by using various options. When you specify any of the options listed below, those sections of the HELP file are made available to you. Therefore, requested options determine what portion of the file is displayed. However, not all of the following option sections are available in all HELP files.

BRIEF, DETAIL, and RELATED are *layering options*, which allow you to specify the level of information. BRIEF, the default, provides concise information to the inexperienced user. DETAIL provides more complete information to the experienced user. RELATED provides the user with additional information about a task.

BRIEF, DETAIL, and RELATED are also conflicting options. You should specify only one of these options in the command string. If you specify more than one of these options at a time, the last option you enter will be in effect.

DETAIL consists of its own *subsetting options*, which allow you to choose a specific part or parts of the help file. They include ALL, DESCript, FORMat, PARMs, NOTES, OPTions, and ERRors. ALL, the default, gives you all of the above-listed subsetting options. You can specify the other options in any combination.

The last set of options is referred to as *other options*. They include SCReen, NOScreen, EXTend, TYPE, and NOType.

OptionA, Layering Options**BRIef**

displays a concise description of the specified command, its basic syntax (command without options), an example, and, if applicable, a message telling the user that either ALL or RELATED information is available.

DETail

displays the detail subset of the HELP information for the specified command. The DETAIL subset can consist of any or all of the following help file sections:

ALL
 DESCript
 FORMat
 PARMs
 OPTions
 NOTEs
 ERRors.

If you specify the DETAIL subset on the command line, it will be in effect for only the current HELP request. If you specify the DETAIL subset by using the DEFAULTS command, it will be in effect for future HELP requests. The default for the detail subset is ALL. If this default has not been changed and if the DETAIL option is specified,

help erase (detail

displays all the HELP information for the ERASE command, excluding the BRIEF and RELATED information. If the default has been changed, for example, to NOTES, and if the detail option is specified,

help erase (detail

displays the Usage Notes information for the ERASE command.

RELated

displays information in a task panel from which you can select help on related topics. For example,

help sendfile (related

displays a task panel from which you can select help on commands related to sendfile:

NAMES
 NOTE
 RDRLIST
 RECEIVE
 SET MSG
 TELL

OptionB, Subsetting Options**ALL**

displays all of the six HELP file sections listed below.

DESCript

displays a general description of the specified HELP file.

FORMat

displays the format section (the syntax).

PARMs

displays the parameter section (explanation of the operands).

OPTions

displays the options section (a list of available options with a brief description).

NOTEs

displays the usage notes and example sections.

ERRors

displays the error message and response sections.

OptionC, Other Options

SCReen

specifies that the entire screen is used to display the HELP file. While viewing the help file you can use PF keys and some System Product Editor commands to manipulate the display. This option is ignored on a line-oriented terminal.

NOScreen

specifies that the file is typed out on the screen.

TYPe

allows error message DMSHLL254E to be issued.

NOType

suppresses the printing of error message DMSHLL254E. This option allows you to change the message's text and placement.

EXTend

uses the HELP search order when you issue the HELP command and the component is specified. If the file is not found in the specified component, the HELP search order is used to locate the HELP file.

Usage Notes

1. HELP is always displayed in a window named HELPWIN that is showing virtual screen HELPWIN. When displaying BRIEF HELP using full-screen CMS, the CMS window is scrolled up, if required, so that you see both your work entry and the HELP information at the same time.
2. You can use the DEFAULTS command to set the HELP options. The initial HELP options have been preset to BRIef, ALL, and SCReen. However, the options you specify on the command line when entering the HELP command override those specified in the DEFAULTS command. This allows you to customize the defaults of the HELP command, yet override them when you desire. For more information, refer to the DEFAULTS command.
3. The HELP disk is specified at system generation time by the system support personnel. If the disk isn't already accessed, HELP accesses the disk containing the system HELP files. The HELP disk is accessed using the last available file mode and remains accessed after HELP has completed processing.
4. For commands or statement names containing special characters, use the special character. If, for example, you wanted to display the HELP file for the EXEC statement &ARG, you would issue HELP EXEC &ARG. For more information, see "Tailoring the HELP Facility," in the *CMS User's Guide*.
5. If you enter 'CP TERM SCROLL nnn' on a line-oriented terminal, it allows you to specify the number of lines to be scrolled on the display screen. For normal frame by frame scrolling, specify 'nnn' to equal the number of data lines on the screen. Specifying 'CONT' instead, causes continuous scrolling to the end of a file.

6. If you request HELP on a line-oriented terminal by issuing the command 'HELP XEDIT MENU' or on a display terminal by issuing 'HELP XEDIT MENU (NOSCREEN)', you get the following:

```
Ready;
help xedit menu (noscreen
For information on one of the following commands,
if its name is preceded by an '*', enter 'HELP name MENU'
    or if preceded by a ':', enter 'HELP name TASK'
    otherwise, enter 'HELP XEDIT name'.
```

&	CDelete	CP	Help	Next	REFRESH	SOS
*Prefix	CFirst	CReplace	HEXType	NFind	RENum	SPLit
*QUERY	Change	CURsor	Input	NFINDUp	REPEat	SPLTJOIN
*SET	CInsert	DELeTe	Join	Overlay	ReplacE	STAck
?	CLAsT	Down	LEft	PARSE	RESet	STATUs
:XEDIT	CLocate	DUPLicat	LOAD	POWerinp	REStorE	TOP
=	CMS	EMSG	Locate	PREServe	RGTLEFT	TRAnSfer
Add	CMSG	EXPanD	LOWercas	PURge	RIght	Type
ALL	CMSXEDIT	EXTRact	LPrefix	PUT	SAVE	Up
ALter	COMMAND	FILE	MACRO	PUTD	SCHANGE	UPPercas
BACKward	COMPRESS	Find	MERge	Query	SET	VMFDEOPT
Bottom	COpy	FINDUp	MODify	QUIT	SHift	VMFOPT
CANCEL	COUnt	FORward	MOve	READ	SI	Xedit
CAppend	COVerlay	GET	MSG	RECOVer	SORT	

```
Ready;
```

If you request help by issuing 'HELP TASK' on a line-oriented terminal or by issuing 'HELP TASK (NOSCREEN)' on a display terminal, you see the following:

```
Ready;
help task (noscreen
For information on one of the following topics, enter
HELP followed by the request information for that topic.
```

Request	Topic	
TASK TASK	TASKS	- Help if you don't know VM commands. Good choice for beginners.
TASK TASK		
MSG HELP	MESSAGE	- Explain how to get help for VM messages.
HELPINFO TASK	HELP	- Explain ways to use and comment on HELP.
MENUS MENU	MENUS	- List the HELP component MENUS.
COMMANDS MENU	COMMANDS	- List VM commands that you can use.
CMS MENU	CMS	- Show only CMS commands.
MACROS	MACROS	- Show CMS assembler language macros.
CP MENU	CP	- Show CP commands for general users.
CPOTHER MENU	CPOTHER	- Show CP commands for other than general users.
XEDIT MENU	XEDIT	- List System Product Editor items.
REXX MENU	REXX	- Help you use the REXX language.
ROUTINE MENU	ROUTINE	- Show routines from callable services library.
RSCS MENU	RSCS	- List Remote Spooling Communications choices.
SQLDS MENU	SQLDS	- Show SQL/Data System items.
SRPI TASK	SRPI	- List the SRPI commands.

```
Ready;
```


HELP

7. If you are viewing a command HELP file, you can issue the MOREHELP command from the command line. The default for MOREHELP is to display the DETAIL layer of that HELP file. DETAIL is determined by your setting of the DEFAULTS options. The options include ALL, DESCript, FORMat, PARMs, OPTions, NOTEs, and ERRors. You can specify an option(s) on the command line to view a certain section(s) of the HELP file. For example, if you are viewing a HELP file and decide that you want to see the format section for that file, you can enter the following on the command line:

```
morehelp (format
```

The format section for that file will be displayed.

8. HELP loads the HELPXED XEDIT macro into user storage (using the EXECLOAD command) if the macro has not already been loaded. Loading this file into storage improves the performance of the HELP command. If you occasionally use HELP, you may want to EXECDROP the HELPXED XEDIT macro after using HELP to release the storage.
9. Any data not within a conditional section (.cs on/.cs off) is uncontrolled data; all uncontrolled data defaults to DETAIL.
10. If a HELP file does not contain any of the requested sections, then the user receives a warning message and alternate information. For more information, see the *VM/SP CMS User's Guide*.
11. Some CMS commands are inappropriate to the BRIEF HELP environment. They are:

```
DELETE WINDOW
DROP WINDOW
HIDE WINDOW
```

Issuing any command that alters, drops, hides, or changes the contents of a BRIEF window may cause undesirable results.

Furthermore, you can use all of the System Product Editor commands while viewing the displayed HELP files except the following:

```
ALL
FILE
INPUT
MACRO
READ
REPLACE
SET
POWERINP
```

Examples

Following are some examples of HELP requests.

- To request a HELP file for *message* DMSHLP002E, issue either:

```
help dms002E
or
help dmshlp002E
```
- To request a *menu* of CP commands, issue:

```
help cp menu
```
- To request a HELP file for the XEDIT LOCATE *subcommand*, issue:

```
help xedit locate
```

- To request a *description* for the CMS TAPE command, issue either:
 help cms tape (desc
 or
 help tape (desc
- If you are viewing the HELP file for the CMS PRINT command and decide you want to get help on the CMS COPYFILE command, issue:
 help cms copyfile
- To request a display of *RELATED* information of the HELP file for the CMS SENDFILE command, issue:
 help cms sendfile (related
- If you are editing a file and want help on the COPYFILE command, and you are not sure whether COPYFILE is an Editor, CP, or CMS command, enter:
 help copyfile (extend
- To request a HELP file for the CMS SET APL command, you may enter:
 help cmsset apl
- To display choices for CMS macros, enter:
 help macros
- To display choices for routines, enter:
 help routines

Messages and Return Codes

DMSHEL241I	Press PF10 for detail information.
DMSHEL241I	Press PF11 to get related information.
DMSHEL241I	Press PF10 for detail information; PF11 to get related information.
DMSHEL242I	This HELP file <i>fn ft</i> has not been converted to the current release format or contains an invalid format word. For information, refer to the CMS User's Guide.
DMSHEL244W	Requested HELP section unavailable; <i>option</i> option assumed.
DMSHEL254E	HELP cannot find the information you requested. If not misspelled, please enter HELP for menu assistance or HELP HELP for the HELP command. [RC = 28]
DMSHEL529E	Subcommand is only valid in display mode
DMSHEL529E	SET <i>commandword</i> subcommand is only valid in editing mode
DMSHEL545E	Missing operand(s)
DMSHEL561E	Cursor is not on a valid data field
DMSHEL586E	String not found
DMSHEL657E	Undefined PFkey/PAkey.
DMSHLC003E	Invalid option: <i>option</i> [RC = 24]
DMSHLL242I	This HELP file <i>fname ftype</i> has not been converted to the current release format or contains an invalid format word. For information, refer to the CMS User's Guide.
DMSHLL243I	RELATED information is not available.
DMSHLL244W	Requested HELP subset information is not available; the <i>option</i> option has been assumed.
DMSHLL254E	HELP cannot find the information you requested. If not misspelled, please enter HELP for menu assistance or HELP HELP for the HELP command. [RC = 28]

HELP

DMSHLL355I For related information on this subject, enter MOREHELP
(RELATED).

DMSHLL356I For more detail on this subject, enter MOREHELP.

DMSHLL639E Error in *routine* routine; return code was *xx*

DMSHLZ242I This HELP file *fname ftype* has not been converted to the current
release format or contains an invalid format word. For
information, refer to the CMS User's Guide.

Additional system messages may be issued by this command. The reasons for these
messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

RELATED section. For further information on how to process these files, please see "Tailoring the HELP Facility" in the *VM/SP CMS User's Guide*.

3. If you create your own HELP files, make sure to store them on minidisks. The HELP facility cannot access HELP files in a file pool.

Messages and Return Codes

DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
 DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
 DMSHLB251E HELP processing error code nnn-description [RC=12]
 DMSHLI002E File not found [RC=28]
 DMSHLI104S Error *nm* reading file *fn ft fm* from disk or directory [RC=104]
 DMSHLI109S Virtual storage capacity exceeded [RC=104]
 DMSHLS109S Virtual storage capacity exceeded [RC=104]
 DMSHLS250S I/O error or device error [RC=256]
 DMSHLP251E HELP processing error code nnn-description [RC=12]
 DMSHLS907E I/O error on file *fn ft fm* [RC=256]

Code Description

801 Numeric format word parameter is outside valid range.
 802 Format word parameter should be a number.
 803 Invalid format word.
 804 Format word parameter missing.
 805 Invalid format word parameter.
 806 Undent greater than indent.
 807 Excessive or negative space count generated.

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

IDENTIFY

Use the IDENTIFY command to display or stack the following information: your user ID and node; the user ID of the RSCS virtual machine; the date, time, time zone, and day of the week.

Format

Identify	<p>[(options...[])]</p> <p><u>Options:</u> [STACK [<u>FIFO</u> LIFO]]</p> <p>[<u>FIFO</u> <u>LIFO</u> <u>TYPE</u>]</p>
-----------------	---

Options

STACK FIFO
LIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked FIFO. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked LIFO. This option is equivalent to STACK LIFO.

TYPE

specifies that the information should be displayed at the terminal. This is the default option.

Example

To display the user ID information at your terminal, you would enter the following:
identify (type

Responses

The following information is displayed or stacked:

```
userid AT node VIA rscsid date time zone day
```

IDENTIFY

Operands

- userid** is the user ID of your virtual machine.
- node** is the RSCS node of your computer.
- rscsid** is the user ID of the RSCS virtual machine.
- date** is the local date, in the form mm/dd/yy.
- time** is the local time, in the form hh:mm:ss.
- zone** is the local time zone.
- day** is the day of the week.

Implementation notes:

The user ID and node are from the CP QUERY USERID command. The date, time, and zone are from the CP QUERY TIME command.

The CP QUERY CPUID command is used to retrieve the CPU serial number. (CP QUERY CPUID returns a 16-digit processor identification; however, IDENTIFY only uses digits three through eight.) This number is then looked up in the file SYSTEM NETID *. That file will have one or more lines of the form:

```
cpu-serial-number node rscsid
```

If there is a conflict in nodes between the SYSTEM NETID file and CP QUERY USERID, the node in SYSTEM NETID takes precedence. If there is no record with a matching serial number, or if the file is not found, the RSCS ID is set to *.

IDENTIFY keeps some of its information in storage, such as user ID, node and RSCS ID. To change any of that information, you must re-IPL CMS and then reissue the IDENTIFY command.

Important Note:

The person responsible for the CMS system at an installation is responsible for creating the SYSTEM NETID file. This file should have a file mode of S2.

Messages and Return Codes

- DMSIDE003E Invalid option: *option* [RC=24]
- DMSIDE056E File *fn ft [fm]* contains invalid record formats [RC=32]
- DMSIDE070E Invalid parameter *parameter* [RC=24]
- DMSIDE104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

IMMCMD

Use the IMMCMD command to establish or cancel Immediate commands from within an exec. IMMCMD determines whether a particular Immediate command has been established or if it has been issued by the terminal user.

Format

IMMCMD	$\left\{ \begin{array}{l} \text{SET} \\ \text{CLEAR} \\ \text{QUERY} \\ \text{STATUS} \end{array} \right\} \quad \textit{name}$
---------------	---

Operands

SET

establishes an Immediate command. If an Immediate command with the same name already exists, it is overridden in a stack-like manner.

CLEAR

clears an Immediate command. Any previously overridden Immediate command with the same name is reinstated by this action.

QUERY

indicates whether the Immediate command has been established. A return code of 0 indicates that the command has been established. A return code of 44 indicates that the command has not been established.

STATUS

indicates whether the command has been issued by the terminal user. The only output from the STATUS operand is a return code of 0 or 1. A return code of 0 indicates that the Immediate command has not been entered from the terminal since the last invocation of the IMMCMD SET or IMMCMD STATUS for that Immediate command. A return code of 1 indicates that the Immediate command has been issued since the last invocation of IMMCMD SET or IMMCMD STATUS for that Immediate command.

name

the 1 to 8 character name of the Immediate command that is established (SET), cancelled (CLEAR) or inquired about (STATUS, QUERY). This operand is always required.

Usage Notes

1. The IMMCMD command should be used only from exec files (CMS exec, EXEC 2, or System Product Interpreter).
2. All Immediate commands established by the IMMCMD command can be explicitly cancelled. If these Immediate commands are not explicitly cancelled by the IMMCMD command, they are cancelled automatically. They are cancelled either by returning to the CMS command environment (if not in CMS subset) or by entry to the CMS abend routine. User exit routines cannot be used with Immediate commands that are established by the IMMCMD command.

IMMCMD

3. Since no exit routine can be given control when an Immediate command (declared via IMMCMD) has been issued from the terminal, the exec writer must use the STATUS operand of IMMCMD. A return code of 1 from IMMCMD STATUS informs the exec writer that a particular Immediate command has been issued by the terminal user. The exec writer can then take appropriate processing action if the Immediate command has been issued.

For a general discussion on Immediate commands, see the *VM/SP CMS User's Guide*. For information on creating Immediate commands, see *VM/SP Application Development Guide for CMS*.

Example

To clear the HM immediate command, you would enter into your exec (written in the REXX language):

```
'immcmd clear hm'
```

Messages and Return Codes

DMSIMM014E	Invalid function <i>function</i> [RC = 24]
DMSIMM047E	No function specified [RC = 24]
DMSIMM070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSIMM109S	Virtual storage capacity exceeded [RC = 104]
DMSIMM261E	No immediate command name was specified [RC = 24]
DMSIMM262E	Immediate command <i>command</i> not found [RC = 44]
DMSIMM263E	Specified immediate command is a nucleus extension and cannot be cleared [RC = 48]

INCLUDE

Use the INCLUDE command to read one or more TEXT files (containing relocatable object code) from a disk or directory and to load them into virtual storage, establishing the proper linkages between the files. A LOAD command must have been previously issued for the INCLUDE command to produce desirable results. For information on the CMS loader and the handling of unresolved references, see the description of the LOAD command.

The INCLUDE command behaves differently depending upon the environment it is issued from. In the following description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. The behavior of the INCLUDE command in both environments is described below to help you plan and develop applications that will run in both environments.

Note: The SET LOADAREA command affects the outcome of the LOAD and INCLUDE commands.

Format

INclude	<p><i>fn...</i> [(options...[])]</p> <p>Options:</p> <p>[CLEAR] [RESET{<i>entry</i>}] [ORIGIN { <i>hexloc</i> }] [NOCLEAR] [*] [TRANS]</p> <p>[MAP] [TYPE] [INV] [REP] [NOMAP] [NOTYPE] [NOINV] [NOREP]</p> <p>[AUTO] [LIBE] [START] [SAME] [NOAUTO] [NOLIBE]</p> <p>[DUP] [NORLDsave] [HIST] [NODUP] [RLDsave] [NOHIST]</p>
----------------	---

Operands

fn...

are the names of the files to be loaded into storage. Files must have a file type of TEXT and consist of relocatable object code such as that produced by the OS language processor. If a GLOBAL TXTLIB command has identified one or more TXTLIBs, *fn* may indicate the name of a TXTLIB member.

Options

If you specified options with a previous LOAD or INCLUDE command, these options (with the exception of CLEAR, NODUP, and ORIGIN) remain set if SAME is specified when INCLUDE is issued. Otherwise, the options assume their default settings. If conflicting options are specified, the last one entered is in effect.

CLEAR

clears the load area in storage to binary zeros before the files are loaded.

INCLUDE

NOCLEAR

does not clear the load area before loading. NOCLEAR is the default.

RESET *entry*

RESET *

resets the execution starting point previously set by a LOAD or INCLUDE command. If *entry* is specified, the starting execution address is reset to the specified location. If an asterisk (*) is specified or if the RESET option is omitted, the loader input is searched for control statements. The entry point is selected from the last ENTRY statement encountered or from an assembler- or compiler-produced END statement. If none is found, a default entry point is selected as follows:

- if you specified an asterisk, the first byte of the first control section loaded by the INCLUDE command becomes the default entry point.
- if you omitted the RESET option, the entry point defaults to the execution starting point previously set by a LOAD or INCLUDE command.

ORIGIN *hexloc*

ORIGIN TRANS

specifies where CMS loads the program. This location must be in the CMS transient area or in any free CMS storage.

ORIGIN *hexloc* loads the program at *hexloc*. This variable is a hexadecimal number of up to eight characters.

ORIGIN TRANS loads the program into the transient area.

If you do not specify the ORIGIN option, CMS loads the program at the first available location past the previously loaded or included text file. INCLUDE does not overlay any previously loaded files unless you specify ORIGIN and specify a location within a previously loaded, non-relocatable object module.

If you specify an ORIGIN address where residency conflicts with the residency of the load as determined by the LOAD command RMODE/ORIGIN option, the INCLUDE will terminate with an error. The following example will cause the INCLUDE to fail:

1. The location of the load, first determined by the LOAD command is above the 16Mb line.
2. You define an address below the 16Mb line in the ORIGIN option of a subsequent INCLUDE.

MAP

adds information to the load map. MAP is the default.

NOMAP

does not add any information to the load map.

TYPE

displays the load map at your terminal and writes it on your disk or directory accessed as A.

NOTYPE

does not display the load map at the terminal. NOTYPE is the default.

INV

writes invalid card images in the LOAD MAP file. INV is the default.

NOINV

does not write invalid card images in the LOAD MAP file.

REP

writes Replace (REP) statement images in the LOAD MAP file. See the explanation of the CMS LOAD command for a description of the Replace (REP) statement. REP is the default.

NOREP

suppresses the writing of Replace (REP) statements in the LOAD MAP file.

AUTO

searches your disks and directories for TEXT files to resolve undefined references. AUTO is the default.

NOAUTO

suppresses automatic searching for TEXT files.

LIBE

searches the text libraries defined by the GLOBAL command for missing subroutines. LIBE is the default.

NOLIBE

does not search any text libraries for unresolved references.

START

begins execution after loading is completed.

SAME

retains the same options (except ORIGIN, NODUP, and CLEAR) that were used by a previous INCLUDE or LOAD command. Otherwise, the default setting of unspecified options is assumed. If other options are specified with SAME, they override previously specified options. (See Usage Note 1..)

DUP

displays warning messages at your virtual console when a duplicate CSECT is encountered during processing. The duplicate CSECT is not loaded. DUP is the default.

NODUP

does not display warning messages at your virtual console when duplicate CSECTs are encountered during processing. The duplicate CSECT is not loaded.

NORLDSav

instructs the CMS LOADER not to save the relocation information from the text files. NORLDSav is the default.

RLDsave

instructs the CMS LOADER to save the relocation information from the text files. The GENMOD command uses relocation information to generate relocatable CMS module files.

HIST

saves the history information (comments) from text files. This information is later included in the module generated by a GENMOD command. The history information from the specified text files is added to history information requested from other text files for this module (using the HIST option on the LOAD command and other INCLUDE commands).

NOHIST

does not save history information from text files. NOHIST is the default.

INCLUDE

Usage Notes

1. If you specify several nondefault options on the LOAD command and you want those options to remain in effect, use the SAME option when you issue the INCLUDE command; for example:

```
include main subi data (reset main map start)
```

brings the files named MAIN TEXT, SUBI TEXT, and DATA TEXT into virtual storage and appends them to previously loaded files. Information about these loaded files is added to the LOAD MAP file. Execution begins at entry point MAIN.

```
load myprog (nomap nolibe norep)
include mysub (map same)
```

During execution of the LOAD command, the file named MYPROG TEXT is brought into real storage. The following options are in effect: NOMAP, NOLIBE, NOREP, NOTYPE, INV, and AUTO. During execution of the INCLUDE command, the file named MYSUB TEXT is appended to MYPROG TEXT. The following options are in effect:

MAP, NOLIBE, NOREP, NOTYPE, INV, AUTO

2. When the INCLUDE command is issued, the loader tables are not reset.
3. When you specify the HIST option, up to 819 comment records can be saved from text files. These records are included later in the module generated by a GENMOD command. When this limit is exceeded, a message is displayed and a warning is placed in the last record of the history data to indicate this condition. Any history information exceeding 819 records is not included.
4. An INCLUDE that intersects a previously loaded nonrelocatable program causes that program to be deleted from storage.

A program loaded by the INCLUDE command that replaces a program loaded by the LOAD command will retain the load attributes.
5. An INCLUDE is not valid after a LOADMOD command. Unpredictable results may occur.
6. Due to the interactive environment of CMS, in an 370-XA mode virtual machine that has greater than 16Mb of storage, you should use caution when following a LOAD command by a number of INCLUDE commands. If no RMODE/ORIGIN option is specified on the LOAD command, and SET LOADAREA RESPECT is in effect, the LOAD process will begin loading according to the RMODE encountered on the first TEXT ESD. If this is RMODE ANY, the load will start above the 16Mb line. Should a more restrictive RMODE be encountered on a subsequent TEXT ESD, then loading will restart below the 16Mb line. Because other commands may be executed between INCLUDE commands, the disks and/or directories used to load the TEXT files specified on the LOAD or previous INCLUDE commands may be released and/or changed. The CMS LOAD process will restart in the environment that is in effect at the point of the restart.
7. Issuing a LOAD command with the RLDSAVE option will result in a relocatable module, even if the RLDSAVE option is not specified on a subsequent INCLUDE command. All RLD data associated with these programs will be saved.
8. If you issue a LOAD command with the NORLDSAV option, and specify RLDSAVE on a subsequent INCLUDE command, the RLD data will be saved

from the time you issued the INCLUDE command. The module will be labeled relocatable, but may not function correctly.

Responses

DMSLIO740I Execution begins ...

START was specified with INCLUDE and the loaded program has begun execution. Any further responses are from the program.

INVALID CARD - xxx...xxx

INV was specified with LOAD and an invalid card has been found. The message and the contents of the invalid card (xxx...xxx) are listed in the LOAD MAP file. The invalid card is ignored and loading continues.

Messages and Return Codes

DMSERD257T Internal system error at address *addr* (offset *offset*)

DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC = 31]

DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*

DMSLIO001E No filename specified [RC = 24]

DMSLIO002E File(s) *fn* TXTLIB not found [RC = 28]

DMSLIO003E Invalid option: *option* [RC = 24]

DMSLIO005E No *option* specified [RC = 24]

DMSLIO021E Entry point *name* not found [RC = 40]

DMSLIO029E Invalid parameter *parameter* in the option *option* field [RC = 24]

DMSLIO055E No entry point defined [RC = 40]

DMSLIO056E File *fn ft [fm]* contains invalid record formats [RC = 32]

DMSLIO099E CMS/DOS environment not active [RC = 40]

DMSLIO104S Error *nn* reading file *fn ft fm* from disk or directory
[RC = 31|55|70|76|99|100]

DMSLIO105S Error *nn* writing file *fn ft fm* on disk or directory
[RC = 31|55|70|76|99|100]

DMSLIO109S Virtual storage capacity exceeded [RC = 104]

DMSLIO116S Loader table overflow [RC = 104]

DMSLIO168S Pseudo register table overflow [RC = 104]

DMSLIO169S ESDID table overflow [RC = 104]

DMSLIO201W The following names are undefined: *namelist* [RC = 4]

DMSLIO202W Duplicate identifier *identifier* [RC = 4]

DMSLIO203W SET LOCATION COUNTER *name* undefined [RC = 4]

DMSLIO206W Pseudo register alignment error [RC = 4]

DMSLIO379E INCLUDE address *at or above|below* 16Mb conflicts with LOAD
address *below|at or above* 16Mb, INCLUDE failed [RC = 88]

DMSLIO380E Storage at origin *addr* in use, *file* not loaded. [RC = 104]

DMSLIO381E Insufficient storage available below 16Mb to load *file*. [RC = 88]

DMSLIO623S Module cannot be loaded at location *location*—this area is available
for system use only [RC = 88]

DMSLIO625S There are too many items that require relocation to save all of the
RLD information [RC = 104]

DMSLIO749W There are too many comments in text files to be included in the
module. [RC = 4]

DMSLIO907T I/O error on file *fn ft fm* [RC = 31|55|70|76|99|256]

DMSLIO994W Restrictive RMODE encountered in CSECT *cname*, LOAD
continues below 16Mb.

DMSLIO997E Running in 370-mode machine and *LOAD|INCLUDE* address is at
or above 16Mb, *LOAD|INCLUDE* failed. [RC = 64]

INCLUDE

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

LABELDEF

Use the LABELDEF command to specify standard HDR1 and EOF1 tape label description information for CMS, CMS/DOS, and OS simulation. This command is required for CMS/DOS and CMS tape label processing. It is optional for OS simulation. However, it is needed if you want to specify a file name to be checked or the exact data to be written in any field of an output HDR1 and EOF1 label.

Format

L Labeldef	<pre> { * } { fn } CLEAR { [FID { ? }] [VOLID { valid ? }] [VOLSEQ volseq] } { [FSEQ fseq] [GENN genn] [GENV genv] } { [CRDTE yyddd] [EXDTE yyddd] [SEC { 0 }] } { [1] } { [3] } [(options...[)]] Options: [PERM] [CHANGE] [NOCHANGE] </pre>
-------------------	---

Operands

*
may be specified only with CLEAR. It clears all existing label definitions.

filename

is one of the following:

ddname for FILEDEF files (OS simulation). When using FORTRAN, you must explicitly specify ddname as FTnnF001.

file name in DTFMT macro (CMS/DOS simulation).

labeldefid specified in the TAPEMAC or TAPPDS command or in the LABID field of the TAPESL macro (can be 1-8 characters).

CLEAR

removes a label definition.

LABELDEF *filename* CLEAR clears only the label definition for that file name.

LABELDEF * CLEAR removes all existing label definitions unless specified as PERM.

FID ?**FID *fid***

supplies the file (data set for OS) identifier in the tape label. Use the FID ? form if the identifier exceeds 8 characters (up to a maximum of 44) or the identifier contains special characters. The system responds by prompting you to supply the information. If the file identifier does not exceed 8 characters, enter the file ID directly (FID *fid*). If the file identifier exceeds 17, only the rightmost 17 characters are moved into the HDR1 record when a file is written to tape, and only the rightmost 17 are displayed for a LABELDEF query.

VOLID *valid***VOLID ?****VOLID SCRATCH**

supplies the volume serial number(s) (1-6 alphanumeric characters). If there is only one volume ID or if you are not using OS simulation, you can enter the volume ID directly (VOLID *valid*).

If you use OS simulation and have multiple volume IDs to process, use the VOLID ? form. The system responds by prompting you to supply the volume serial numbers needed to process the file.

For non-OS simulation, more than one volume ID may be specified; however, only the first volume ID will be used.

Specifying a volume ID of SCRATCH results in no serial number checking for that tape and for any subsequent tapes to be mounted for the current file. When specifying multiple volume IDs, SCRATCH must be the last volume ID specified. If you specify a VOLID of SCRATCH and you are a non-OS simulation user, a tape with the VOLID of 'SCRATC' is searched for.

VOLSEQ *volseq*

is the volume sequence number (1-4 numeric characters). Under OS-simulation the volume sequence number is ignored and is set to 0001.

FSEQ *fseq*

is the file (data set for OS) sequence number in the label (1-4 numeric characters).

GENN *genn*

is the generation number (1-4 numeric characters).

GENV *genv*

is the generation version (1-2 numeric characters).

CRDTE *yyddd*

is the creation date.

EXDTE *yyddd*

is the expiration date.

SEC

specifies security classification (0, 1, or 3). See the IBM publication *OS/VS Tape Labels*, for the meaning of security classification on tape files. Note that this number has no effect on how the file is processed. It is used only for checking or writing purposes.

Options**PERM**

retains the current definition until it either is explicitly cleared or is changed by a new LABELDEF command with the CHANGE option. If PERM is not specified, the definition is cleared when a LABELDEF * CLEAR command is executed.

CHANGE

merges the label definitions whenever a label definition already exists for a file name and a new LABELDEF command specifying the same file name is issued. In this situation, the options associated with the two definitions are merged. Options from the original definition remain in effect unless duplicated in the new definition. New options are added to the option list.

NOCHANGE

retains the current label definition, if one exists, for the specified file name.

The following default values are used in output labels when a value is not explicitly specified:

FID

For OS simulation, fid is the ddname specified in the FILEDEF command for the file.

For CMS/DOS, fid is the DTFMT symbolic name.

For the CMS TAPESL macro, fid is the LABELDEF specified in the LABID parameter.

VOLID

is CMS001. For OS simulation, the actual VOLID from the tape mounted is used if processing an "SL" tape file.

FSEQ

is 0001.

VOLSEQ

is 0001.

GENN

is blanks.

GENV

is blanks.

CRDTE

is the date when the label is written.

EXDTE

is the date when the label is written.

SEC

is 0.

Usage Notes

1. To check a field in an input label, specify it on your LABELDEF command for the label. If you do not specify a value for a particular field, this field is not checked at all for input. For output, any field you specify is written in the label exactly as you specify it on the LABELDEF command. If you do not specify a field for output, the default value for that field is written in the label.

If you write the following LABELDEF command,

```
labeldef filex fid master fseq 2 exdte 78285
```

and use the statement for an input file, only the file identifier, file sequence number, and expiration date in HDR1 labels are checked. Error messages are issued when there fields in the tape label do not match those specified in the LABELDEF statement. If you use the same statement for an output file, the fields leave the following values:

file ID	MASTER
file sequence number	0002
volume sequence number	0001
creation date	date when label is written
expiration date	78285
security	0
volume serial number	CMS001
generation number	blank
generation version	blank

2. If you issue LABELDEF without any operands, a list of all labeldefs currently in effect is displayed on your terminal. For an OS simulation user, if SCRATCH was entered at command time and the file has not been opened, then all the volume IDs and SCRATCH will be displayed. If SCRATCH was entered at command time and the file has been opened, then all the volume IDs and the volume IDs of all the scratch tapes will be displayed following SCRATC or SCRATCH.
3. For OS simulation, a LABELDEF statement may be used as well as a FILEDEF statement for a file. Use of a LABELDEF statement is optional in this case. The statements

```
labeldef filez fid payroll fseq 2 exdte 78300
filedef filez tap1 sl volid vol4
```

define filez as a labeled tape file on tape 181. The volume serial is VOL4, the fileid is PAYROLL, and the file sequence number is 0002. Expiration date is day 300 in 1978. If you only use the FILEDEF command, you have only defined the volume ID (volume serial number).

4. For CMS and CMS/DOS, a LABELDEF command is required. The command

```
labeldef file14 volid supvol volseq 3
```

defines a tape label with a volume serial of SUPVOL and a volume sequence number of 0003. This LABELDEF statement could be used by a CMS/DOS program containing a DTFMT macro with the form

```
FILE14 DTFMT ...FILABL=STD...
```

or by a CMS program with a TAPESL macro similar to the following:

```
TAPESL HOUT,181,LABID=FILE14
```

A CMS TAPEMAC command could use the same LABELDEF as follows:

```
tapemac maclib sl file14
```

In all three preceding examples, the LABELDEF statement must be issued before the program or command is executed.

5. See the section “Tape Labels in CMS” in the *VM/SP CMS User's Guide* for more details on CMS tape label processing.

Messages and Return Codes

DMSLBD003E	Invalid option: <i>option</i> [RC = 24]
DMSLBD029E	Invalid parameter <i>parameter</i> in the option <i>option</i> field [RC = 24]
DMSLBD065E	<i>option</i> option specified twice [RC = 24]
DMSLBD066E	<i>option1</i> and <i>option2</i> are conflicting options [RC = 24]
DMSLBD070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSLBD109S	Virtual storage capacity exceeded [RC = 104]
DMSLBD220R	Enter data set name:
DMSLBD221E	Invalid data set name [RC = 24]
DMSLBD324I	No user defined LABELDEFs in effect
DMSLBD438E	Valid <i>valid</i> is a duplicate entry [RC = 24]
DMSLBD439E	Valid <i>valid</i> is an invalid entry [RC = 24]
DMSLBD441R	Enter valid information
DMSLBD442E	SCRATCH may only be used as the last valid for the file [RC = 24]
DMSLBD704I	Invalid CLEAR request

LISTDIR

Use the LISTDIR command to list Shared File System (SFS) directories for a specified directory structure.

Format

LISTDIR	$[\textit{dirid}] \quad [(\text{options...} [])]$												
	Options: <table style="display: inline-table; border: none; vertical-align: middle;"> <tr> <td style="border: 1px solid black; padding: 2px;">ACCEssed</td> <td style="border: 1px solid black; padding: 2px;">STACK</td> <td style="border: 1px solid black; padding: 2px;">FIFO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">ALL</td> <td style="border: 1px solid black; padding: 2px;"></td> <td style="border: 1px solid black; padding: 2px;">LIFO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">SUBdirectory</td> <td style="border: 1px solid black; padding: 2px;">LIFO</td> <td style="border: 1px solid black; padding: 2px;">FIFO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">NOSubdirectory</td> <td style="border: 1px solid black; padding: 2px;">XEDIT</td> <td style="border: 1px solid black; padding: 2px;"></td> </tr> </table>	ACCEssed	STACK	FIFO	ALL		LIFO	SUBdirectory	LIFO	FIFO	NOSubdirectory	XEDIT	
ACCEssed	STACK	FIFO											
ALL		LIFO											
SUBdirectory	LIFO	FIFO											
NOSubdirectory	XEDIT												

Operands*dirid*

identifies the directory where the list begins. The specified directory and its subdirectories for which you have read or write authority are listed. If *dirid* is not specified, the list begins with your top directory. See the ACCESSED option below for an exception to this rule.

See "Naming Shared File System (SFS) Directories" on page 4 for more information on *dirid*.

Options**ACCesed**

lists only the accessed directories in the specified directory structure. If *dirid* is not specified with this option, all accessed directories are listed in CMS search order.

ALL

lists all directories in the specified directory structure for which you have read or write authority, whether they are accessed or not. ALL is the default.

SUBdirectory

lists the directory specified by *dirid* and its subdirectories. The specified directory can not be open when using this operand. SUBDIRECTORY is the default.

NOSUBdirectory

lists only the directory specified by *dirid*. Use this option to test for the existence of a directory. No subdirectories are listed.

If the ACCESSED option is specified with NOSUBDIRECTORY and no *dirid* is specified, the first directory that is accessed in the CMS search order is listed.

STACK [FIFO]**STACK LIFO**

places the information in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

XEDIT

places the list in the file that is currently displayed. This option is only valid when issued from the XEDIT environment. The list replaces the information in the file starting with the current line and continues until the end of the list is reached. Remaining text in the file, if any, is unchanged.

Your edited file must either be a fixed format with a logical record length, LRECL, of 319 or a variable record length format.

Usage Notes

You can invoke the LISTDIR command from the command line, from an exec, or as a function from a program. No error messages are issued if LISTDIR is invoked:

- As a function from a program
- From a CMS exec file that has the &CONTROL NOMSG option in effect
- From an EXEC2 exec where CMDCALL is not in effect
- From a System Product Interpreter exec with ADDRESS COMMAND in effect

Example

The following is an example of LISTDIR:

```
Fm Directory Name
A FILEPOOL:SMITH
- FILEPOOL:SMITH.ADDRESSES
- FILEPOOL:SMITH.ADDRESSES.EMPLOYEES
B FILEPOOL:SMITH.ADDRESSES.MANAGERS
- FILEPOOL:SMITH.BIRTHDAYS
- FILEPOOL:SMITH.BIRTHDAYS.EMPLOYEES
```

Response

The following information is displayed at the terminal unless the STACK, FIFO, or LIFO option is specified:

```
Fm Directory Name
fm dirname
. .
. .
```

where

fm

is the file mode if the directory is accessed. If the directory is not accessed, this column contains a dash (-).

dirname

is the complete name of the directory.

Note: The heading is included only in terminal output.

Messages and Return Codes

DMSJED069E Filemode *mode* not accessed [RC = 36]
 DMSJED109S Virtual storage capacity exceeded [RC = 104]
 DMSJED1187E Too many subdirectory levels in *dirid* [RC = 24]
 DMSJED1188E Filemode *mode* is not associated with a directory [RC = 74]
 DMSJED1189E Filemode *mode* is associated with a top directory [RC = 24]
 DMSJED1223E There is no default file pool currently defined [RC = 40]
 DMSJLD065E *option* option specified twice [RC = 24]
 DMSJLD066E *option* and *option* are conflicting options [RC = 24]
 DMSJLD689E File must be F-format 319 or V-format [RC = 24]
 DMSJLD1160E Directory *dirname* already open. [RC = 70]
 DMSJLD1184E Directory *dirname* not found or you are not authorized for it
 [RC = 28]
 DMSJLD1210E Directory *dirname* not found [RC = 28]
 DMSOUT065E *option* option specified twice [RC = 24]
 DMSOUT066E *option1* and *option2* are conflicting options [RC = 24]
 DMSOUT105S Error *nn* writing file to XEDIT [RC = 100]
 DMSOUT109S Virtual storage capacity exceeded [RC = 104]
 DMSOUT688E XEDIT option only valid from XEDIT environment [RC = 24]
 DMSOUT1201E STACK option cannot follow FIFO or LIFO [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

LISTDS

Use the LISTDS command to list, at your terminal, information about the data sets or files residing on accessed OS or DOS disks. In addition, use LISTDS to display extent or free space information when you want to allocate space for VSAM files.

Format

LISTDS	$\left[\begin{array}{c} ? \\ dsname \end{array} \right]$	$\left\{ \begin{array}{c} fm \\ * \end{array} \right\}$	$[(options... [])]$
	<u>Options:</u>	$[\text{EXTENT}]$	$[\text{FORMAT}]$ $[\text{PDS}]$
		$[\text{FREE}]$	

Operands

?

indicates that you want to enter interactively the OS data set name, VSE file ID, or VSAM data space name. When you enter a question mark (?), CMS prompts you to enter the OS data set name, DOS file ID, or VSAM data space name exactly as it appears on the disk. This form allows you to enter names that contain embedded blanks or hyphens.

dsname

is the OS data set name or VSE file ID or VSAM data space name. It takes the form:

```
qual1 [qual2...qualn]
-- or --
qual1 [.qual2...qualn]
```

where qual1, qual2, through qualn are the qualifiers of the dataset. If blanks separate the qualifiers, the dataset name used will be the concatenation of the qualifiers with periods. (See Usage Note 1.)

fm

is the file mode of the disk to be searched for the specified file. If a dsname is not specified, a list of all the files or data sets on the specified disk is displayed.

*

indicates that you want all of your accessed DOS or OS disks searched for the specified data set or file. If a dsname is not specified, a list of all files on all accessed OS and DOS disks is displayed. If a dsname is specified, CMS stops searching all of your accessed DOS or OS disks as soon as it finds the first copy of the specified data set or file.

Options

The FREE and EXTENT options are mutually exclusive; the FORMAT and PDS options cannot be specified with either FREE or EXTENT.

FREE

requests a display of all free space extents on a specific minidisk or on all accessed DOS and OS disks. If you enter the FREE option, you cannot specify a dsname.

EXTENT**EX**

requests a display of allocated extents for a single file or for an entire disk or minidisk. If a dsname is specified, only the extents for that particular file or data set are listed. If fm is specified as *, all disks are searched for extents occupied by that file, but only the extents for the first file or data set encountered are displayed.

If a dsname is not specified, then a list of all currently allocated extents on the specified disk, or on all disks, is displayed.

FORMAT**FO**

requests a display of the date, disk label, file mode, and data set name for an OS data set as well as RECFM, LRECL, BLKSIZE, and DSORG information. For a VSE file, LISTDS displays the date, disk label, file mode, and file ID, but gives no information about the RECFM, LRECL, and BLKSIZE (two blanks appear for each); DSORG is always PS.

PDS

displays the member names of referenced OS partitioned data sets.

For examples of the displays produced as a result of each of these options, see the "Responses" section, below.

Usage Notes

1. If you want to enter an OS or VSE file identification on the LISTDS command line, it may consist of qualifiers separated by periods or blanks. For example, the file TEST.INPUT.SOURCE.D could be listed as follows:

```
listds test input source d
-- or --
listds test.input.source.d
```

Or, you can enter the name interactively, as follows:

```
listds ? *
DMSLDS220R Enter data set name:
test.input.source.d
```

Note that when the data set name is entered interactively, it must be entered in its exact form; when entered on the LISTDS command line, the periods may be omitted.

You must use the interactive form to enter a VSE file ID that contains embedded blanks.

2. When using access method services, use the FREE option to determine what free space is available for allocation by VSAM. For example:

```
listds * (free
```

requests a display of unallocated extents on all accessed OS or DOS disks. You can then use the EXTENT option on the DLBL command when you define the file for AMSERV.

3. Full disk displays using the FREE option will display free alternate tracks as well as free space extents.
4. Since CMS does not support ISAM files, LISTDS lists extent and free information on ISAM files, but ignores format 2 DSCB's.
5. Since CMS does not support track overflow, LISTDS will not read beyond a track if DCB = RECFM = T is specified for the OS VTOC.
6. LISTDS uses the extended plist for processing the *dsname* parameter. If you are calling LISTDS from an assembler language program and using *dsname*, you should supply an extended plist. *VM/SP Application Development Guide for CMS* has more information on how an assembler language program can supply an extended plist.

Responses

DMSLDS220R Enter data set name:

This message prompts you to enter the data set name when you use the ? operand on the LISTDS command. Enter the file identification in its exact form. A sample sequence might be:

```
listds ? c
DMSLDS220R Enter data set name:
my.file.test
FM DATA SET NAME
C MY.FILE.TEST
Ready;
```

The response shown above following the entry of the data set name is the same as the response given when you enter a data set name on the LISTDS command line.

DMSLDS229I No members found

This message is displayed when you use the PDS option and the data set has no members.

DMSLDS233I No free space available on *fm* disk

This message is displayed when you use the FREE option and there is no free space available on the specified disk.

Responses to the EXTENT Option: A sample response to the EXTENT option is shown below. The headers and the type of information supplied are the same when you request information for a specific file only, or for all disks.

LISTDS

listds g (extent

```
EXTENT INFORMATION FOR 'VTOC' ON 'G' DISK:
SEQ TYPE  CYL-HD(RELTRK) TO  CYL-HD(RELTRK)  TRACKS
000 VTOC  0099 00   1881    0099 18   1899      19
```

```
EXTENT INFORMATION FOR 'PRIVAT.CORE.IMAGE.LIB' ON 'G' DISK:
SEQ TYPE  CYL-HD(RELTRK) TO  CYL-HD(RELTRK)  TRACKS
000 DATA 0000 01     1     0049 18   949      949
```

```
EXTENT INFORMATION FOR 'SYSTEM.WORK.FILE.NO.6' ON 'G' DISK:
SEQ TYPE  CYL-HD(RELTRK) TO  CYL-HD(RELTRK)  TRACKS
000 DATA 0050 00   950    0051 18   987      38
```

```
EXTENT INFORMATION FOR 'COBOL TEST PROGRAM' ON 'G' DISK:
SEQ TYPE  CYL-HD(RELTRK) TO  CYL-HD(RELTRK)  TRACKS
000 DATA 0052 02   990    0054 01  1027     38
```

```
EXTENT INFORMATION FOR 'DKSQ01A' ON 'G' DISK:
SEQ TYPE  CYL-HD(RELTRK) TO  CYL-HD(RELTRK)  TRACKS
000 DATA 0080 01  1521    0081 00  1539     19
```

or for a fixed-block device:

```
EXTENT INFORMATION FOR 'DSQ01A' ON 'G' DISK:
SEQ TYPE  BLOCKNO  TO  BLOCKNO  BLOCKS
000 DATA 000500      000550      51
```

where:

SEQ

indicates the sequence number assigned this extent when the extents were defined via the DLBL command. CMS assigns the sequence numbers for VSAM data sets; the first extent set has a sequence of 000, the second extent has a sequence of 001, and so on.

TYPE

can have the following designations:

Type	Meaning
DATA	Data area extent
VTOC	VTOC extent of the disk
SPLIT	Split cylinder extent
LABEL	User label extent
INDEX	ISAM index area extent
OVFLO	ISAM independent overflow area extent
MODEL	Model data set label in the VTOC. Does not define an extent

CYL-HD(RELTRK) TO CYL-HD(RELTRK)

indicates the cylinder, head, and relative track numbers of the start and end tracks of this extent.

TRACKS

indicates the number of tracks in the extent.

BLOCKNO TO BLOCKNO

indicates the relative block numbers of the start and end of the extent.

BLOCKS

indicates the number of blocks in the extent.

Response to the FREE Option: A sample response to the FREE option is shown below. The same headers and type of information is shown when you request free information for all accessed disks.

```
listds g (free
FREESPACE EXTENTS FOR 'G' DISK:
  CYL-HD(RELTRK) TO  CYL-HD(RELTRK)  TRACKS
0052 00   988      0052 01   989          2
0054 02  1028      0080 00  1520         493
0081 01  1540      0098 18  1880         341
```

or for a fixed-block device:

```
listds g (free
FREESPACE EXTENTS FOR 'G' DISK:
FB/E BLOCKNO  TO  FB/E BLOCKNO  BLOCKS
   000501      001330      830
   010310      029610     19301
   068990      069990     1001
```

where:

CYL-HD(RELTRK) TO CYL-HD(RELTRK)

indicates the cylinder, head and relative track numbers of the starting and ending track in the free extent.

TRACKS

indicates the total number of free tracks in the extent.

BLOCKNO TO BLOCKNO

indicates the relative block number of the start and end of extents that are free on the fixed-block device.

BLOCKS

indicates the total number of blocks contained in each extent.

Response to the FORMAT and PDS Options:

If you enter the FORMAT and PDS options, you receive information similar to the following:

```
listds d (fo pds)

RECFM LRECL BLKSI DSORG  DATE   LABEL  FM  DATA SET NAME
  FB   80   800    PO  01/31/75  OSSYS1  D  SYS1.MACLIB
MEMBER NAMES:
ABEND  ATTACH  BLDL   BSP    CLOSE  DCB  DETACH  DEVTYPE
FIND   PUT     READ  WRITE  XDAP
RECFM LRECL BLKSI DSORG  DATE   LABEL  FM  DATA SET NAME
  F    80    80    PS  01/10/75  OSSYS1  D  SAMPLE
```

Messages and Return Codes

DMSLDS002E Dataset not found [RC=28]
DMSLDS003E Invalid option: *option* [RC=24]
DMSLDS048E Invalid mode *mode* [RC=24]
DMSLDS069E Filemode *mode* not accessed [RC=36]
DMSLDS221E Invalid data set name [RC=24]
DMSLDS222E I/O error reading *data set name* from {*fm*|OS|DOS} disk [RC=28]
DMSLDS223E No filemode specified [RC=24]
DMSLDS226E No dataset name allowed with FREE option [RC=24]
DMSLDS227W Invalid extent found for *data set name* on *fm* disk [RC=4]
DMSLDS231E I/O error reading VTOC from {*fm*|OS|DOS} disk [RC=28]
DMSLDS333E *nnnnnK* partition too large for this virtual machine [RC=24]

LISTFILE

Use the LISTFILE command to obtain specified information about:

- CMS files residing on accessed minidisks
- files and subdirectories in Shared File System (SFS) directories.

Format

Listfile	$\left[\begin{array}{c} fn \\ * \end{array} \left[\begin{array}{c} ft \\ * \end{array} \left[\begin{array}{c} fm \\ * \end{array} \right] \right] \right] \left[(\text{options... } []) \right]$ <p><u>Options:</u> $\left[\begin{array}{c} \text{ALLFile} \\ \text{AUTHfile} \end{array} \right]$ $\left[\begin{array}{c} \text{SHAre} \\ \text{SEArch} \end{array} \right]$ $\left[\begin{array}{c} \text{Header} \\ \text{NOHeader} \end{array} \right]$</p> <table style="width: 100%; border: none;"> <tr> <td style="border: none; padding: 5px;">$\left[\begin{array}{l} \text{Exec} \quad \left[\begin{array}{c} \text{Trace} \\ \text{ARGS} \end{array} \right] \left[\text{ARGS} \right] \\ \text{Trace} \quad \left[\text{ARGS} \right] \\ \text{Append} \quad \left[\text{ARGS} \right] \\ \text{STACK} \quad \left[\text{FIFO} \mid \text{LIFO} \right] \\ \text{FIFO} \\ \text{LIFO} \\ \text{XEDIT} \end{array} \right]$</td> <td style="border: none; padding: 5px;">$\left[\begin{array}{c} \text{FName} \\ \text{FType} \\ \text{FMode} \\ \text{FFormat} \\ \text{ALloc} \\ \text{Date} \\ \text{Label} \end{array} \right]$</td> <td style="border: none; padding: 5px;">$\left[\begin{array}{c} \text{Blocks} \\ \% x \end{array} \right]$</td> </tr> </table>	$\left[\begin{array}{l} \text{Exec} \quad \left[\begin{array}{c} \text{Trace} \\ \text{ARGS} \end{array} \right] \left[\text{ARGS} \right] \\ \text{Trace} \quad \left[\text{ARGS} \right] \\ \text{Append} \quad \left[\text{ARGS} \right] \\ \text{STACK} \quad \left[\text{FIFO} \mid \text{LIFO} \right] \\ \text{FIFO} \\ \text{LIFO} \\ \text{XEDIT} \end{array} \right]$	$\left[\begin{array}{c} \text{FName} \\ \text{FType} \\ \text{FMode} \\ \text{FFormat} \\ \text{ALloc} \\ \text{Date} \\ \text{Label} \end{array} \right]$	$\left[\begin{array}{c} \text{Blocks} \\ \% x \end{array} \right]$
$\left[\begin{array}{l} \text{Exec} \quad \left[\begin{array}{c} \text{Trace} \\ \text{ARGS} \end{array} \right] \left[\text{ARGS} \right] \\ \text{Trace} \quad \left[\text{ARGS} \right] \\ \text{Append} \quad \left[\text{ARGS} \right] \\ \text{STACK} \quad \left[\text{FIFO} \mid \text{LIFO} \right] \\ \text{FIFO} \\ \text{LIFO} \\ \text{XEDIT} \end{array} \right]$	$\left[\begin{array}{c} \text{FName} \\ \text{FType} \\ \text{FMode} \\ \text{FFormat} \\ \text{ALloc} \\ \text{Date} \\ \text{Label} \end{array} \right]$	$\left[\begin{array}{c} \text{Blocks} \\ \% x \end{array} \right]$		

Operands

fn

is the name of the file for which information is to be collected. If an asterisk (*) is coded in this field, all file names are used.

In addition, certain special characters (* and %) can be used as part of the file name to request that the list contain a specific subset of files. See "Pattern Matching" on page 8 for information on using these special characters.

ft

is the file type of the file for which information is to be collected. If an asterisk is coded in this field, all file types are used.

In addition, certain special characters (* and %) can be used as part of the file type to request that the list contain a specific subset of files. See "Pattern Matching" on page 8 for information on using these special characters.

fm

is the file mode of the files for which information is to be collected. If this field is omitted, only the disk or directory accessed as A is searched. If an asterisk is coded, all accessed disks and directories are searched.

Shared File System (SFS) Options

You can use the ALLFILE, AUTHFILE, SHARE, and SEARCH options to specify the information you want about files and subdirectories in SFS directories.

ALLFILE and AUTHFILE are ignored if the specified file(s) is on a disk.

SEARCH cannot be used for files on disks. The SHARE option can be used for files in both directories and on disks.

ALLFile

lists the specified files in a directory whether or not you have been granted read or write authority on them. Subdirectories and erased or revoked aliases are also listed.

AUThfile

lists only the specified files in a directory that you have been granted read or write authority to. Subdirectories and erased or revoked aliases are not listed. AUTHFILE is the default.

SHAre

lists the following information about the specified files (or subdirectory if the ALLFILE option is specified):

- file or directory identifier
- owner
- type (minidisk, directory, base, alias, erased or revoked alias)
- what authority you have to the object (R/W).

The SHARE option cannot be specified with the following options:

- FORMAT
- ALLOC
- DATE
- LABEL
- SEARCH

SEARch

searches SFS directory structures for the specified file(s), and lists the following information:

- file identifier
- directory name

This option is useful if you know the file name, but not the name of the directory where the file resides. You cannot specify an asterisk (*) for the file mode when you use this option.

The specified directory cannot be open when using this option.

The search begins at the directory specified by *fm*. If you have read or write authority to any subdirectories, they are searched also; they do not have to be accessed to be searched. Subdirectories that are locked EXCLUSIVE by another user are not searched. If you do not specify *fm*, the search begins at your top directory whether it is accessed or not accessed.

The SEARCH option cannot be specified with the following options:

- FORMAT
- ALLOC
- DATE
- LABEL
- SHARE
- EXEC
- TRACE
- APPEND
- ARGS

Output Format Options

Header

includes column headings in the listing. **HEADER** is the default if any of the supplemental information options (**FORMAT**, **ALLOCATE**, **SHARE**, **SEARCH**, **DATE**, or **LABEL**) are specified.

NOHeader

does not include column headings in the list. **NOHEADER** is the default if only file name, file type, and/or file mode information is requested, or if you specify **STACK**, **FIFO**, or **LIFO**.

Output Disposition Options

Exec

creates a CMS EXEC file of 80- or 88-character records (one record for each of the files that satisfies the given file identifier) on your disk or directory accessed as A. An 80-character record file is created unless you specify the **LABEL** option, in which case an 88-character record file is created. If a CMS EXEC already exists, it is replaced. The header is not included in the file. This option is not valid with the **SEARCH** option.

Trace

causes the EXEC 2 statement **&TRACE OFF** to be written as the first record of the CMS EXEC file, which is created when the EXEC option is specified. With this option, no statements issued from the CMS EXEC file are traced. For more information on the **&TRACE** statement, see the *VM/SP EXEC 2 Reference*. The **TRACE** option implies the EXEC option. This option is not valid with the **SEARCH** option.

ARGS

causes EXEC 2 dummy arguments **&3** through **&15** to be appended to each line in the CMS EXEC file (following the file ID of each file). Each record of the CMS EXEC file has the form:

```
&1 &2 fileid &3 &4 &5 &6 ...&15
```

Specifying this option allows you to pass up to 15 arguments to the CMS EXEC file. The **ARGS** option does not imply the EXEC option and therefore must be specified in conjunction with EXEC, TRACE, or APPEND. This option is not valid with the **SEARCH** option.

APpend

creates a CMS EXEC and appends it to the existing CMS EXEC file. If no CMS EXEC file exists, one is created. This option is not valid with the **SEARCH** option.

STACK **[FIFO]**

STACK LIFO

specifies that the information should be placed in the program stack (for use by an exec or other program) instead of being displayed at the terminal. The information is stacked either **FIFO** (first in first out) or **LIFO** (last in first out). The default order is **FIFO**.

FIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked **FIFO**. The options **STACK**, **STACK FIFO**, and **FIFO** are all equivalent.

LIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked LIFO. This option is equivalent to **STACK LIFO**.

XEDIT

places the list in the file that is currently displayed. This option is only valid when issued from the **XEDIT** environment. The list replaces the information in the file starting with the current line and continues until the end of the list is reached. Remaining text in the file, if any, is unchanged. The edited file must either be fixed format with a logical record length (**lrecl**) of 108 or variable length format. Each line is 108 characters and contains the information that is obtained with the **LABEL** option, plus the file name, file type, file mode appended to the end of the line.

If both the **XEDIT** and **SHARE** option are specified, the edited file must be either variable length or fixed length with a logical record length (**lrecl**) of 149. Each line is 149 characters and contains the information that is obtained with the **SHARE** option (file name, file type, file mode, owner, type, authority), plus the file name, file type, file mode, records, blocks, date, and time appended to the end of the line.

When the **SEARCH** option is used with the **XEDIT** option, the edited file must be either variable length or fixed length with a logical record length (**lrecl**) of 355. Each line is 355 characters and contains the information that is obtained with the **SEARCH** option (file name, file type, file mode, directory name), plus the file name, file type, file mode, and directory name appended to the end of the line.

Information Request Options

Only one of these options need be specified. If one is specified, any options with a higher priority are also in effect. If none of the following options are specified, the default information request options are in effect.

Default Information Request Options**FName**

creates a list containing only file names. Option priority is 7.

FType

creates a list containing only file names and file types. Option priority is 6.

FMode

creates a list containing file names, file types, and file modes. Option priority is 5.

Supplemental Information Options**FOrmat**

includes the record format and logical record length of each file in the list. Option priority is 4.

ALloc

includes the amount of disk space that CMS has allocated to the specified file in the list. The quantities given are the number of logical records in the file and the number of CMS data blocks. Option priority is 3.

Date

includes the date the file was last written in the list. The form of the date is:
month/day/year hour:minute

for 800-byte block disks, or:

month/day/year hour:minute:second

for all other format sizes. Option priority is 2.

Label

includes the label of the disk on which the file resides in the list. Option priority is 1.

Other Options

Blocks

causes the total number of CMS data blocks used by the files in the list to be displayed as the last line of the list, in the form **BLOCKS n**. It is displayed as a separate line. When **BLOCKS** is specified along with the **STACK** option, the line is displayed and not stacked.

%x

is used to change the place holding character from % to x, where x is any character, for this invocation of LISTFILE. For more information on using a place holding character, see "Pattern Matching" on page 8.

Usage Notes

1. If you enter the LISTFILE command with no operands, a list of all files on your disk or directory accessed as A is displayed at the terminal.
2. If special characters (* or %) are used for *fn* or *ft* and the files are in an SFS directory, the specified directory cannot be open.
3. When pattern matching is done on subdirectory names that contain more than eight characters, the first eight characters are used as the file name and the remaining characters are used as the file type. For example, your directory accessed as A contains,

```
CROCKETT NOTEBOOK
CROCKETTNOTES
```

where CROCKETT NOTEBOOK is a file and CROCKETTNOTES is a subdirectory. Issuing,

```
listfile croc* n* a (allfile
```

would find both CROCKETT NOTEBOOK and CROCKETTNOTES because CROCKETTNOTES contains more than eight characters and is matched as if it had a file name of CROCKETT and a file type of NOTES. Issuing,

```
listfile crockettnotes * a (allfile
```

would also find both CROCKETT NOTEBOOK and CROCKETTNOTES because the LISTFILE command only uses the first eight characters as the file name. Therefore, any file with the name of CROCKETT, or any subdirectory with CROCKETT as the first eight characters in its name would be listed.

4. The default place holding character (%) can be changed by using the %x option. For example,


```
listfile $ script (%$
```

displays all SCRIPT files on the disk or directory accessed as A whose file name is one character in length.
5. If you request any additional information with the supplemental information options, that information is displayed along with the header.

6. When you use the EXEC or APPEND option, the CMS EXEC A1 that is created is in the format:

```
&1 &2 filename filetype filemode
```

where column 1 is blank.

If you specify the ARGS option with EXEC or APPEND, each line in the CMS EXEC is in the format:

```
&1 &2 filename filetype filemode &3 &4 &5 &6 ...&15
```

This allows you to pass up to 15 arguments to the EXEC. For example, if the following command is issued,

```
LISTFILE * * A (EXEC ARGS
```

a CMS EXEC file is created, with each record formatted as shown above. The following command

```
cms tape dump ( wtm
```

causes the tape dumping command to be executed against each file in the CMS EXEC, with TAPE assigned to &1, DUMP to &2, (to &3, and WTM to &4.

If you use any of the supplemental information options, that information is included in the EXEC file. For information on using CMS EXEC files, see the *CMS User's Guide*.

7. You can invoke the LISTFILE command from the command line, from an exec, or as a function from a program. No error messages are issued if LISTFILE is invoked:

- As a function from a program
- From a CMS exec file that has the &CONTROL NOMSG option in effect
- From an EXEC2 exec where CMDCALL is not in effect
- From a System Product Interpreter exec with ADDRESS COMMAND in effect.

8. To display only the files with a particular file mode number, specify the numeric portion of the file mode in the LISTFILE command. For example, to display only the files with file type 'EXEC' and a file mode of A2:

```
Listfile * exec a2
```

The display might look like this:

```
ALPHA EXEC A2
SEND EXEC A2
TEMP EXEC A2
```

9. The options STACK, LIFO, and FIFO cause the requested information to be placed in the program stack. When the requested information is to be stacked, the options relating to the CMS EXEC (APPEND, EXEC, TRACE, and ARGS) and the options relating to the display format (HEADER, NOHEADER) should not be specified.

Responses

Unless the EXEC, TRACE, APPEND, STACK, LIFO, or FIFO option is specified, the requested information is displayed at the terminal. With the exception of the SHARE and SEARCH options and depending on the other options specified, as discussed above, the information displayed is:

FILENAME	FILETYPE	FM	FORMAT	LRECL	RECS	BLOCKS	DATE	TIME	LABEL
fn	ft	fm	format	lrecl	norecs	noblks	mm/dd/yy	hh:mm:ss	valid
.
.
.

where:

fn

is the name of the file or directory.

ft

is the file type of the file. For a directory this column is blank.

fm

is the file mode of the file or directory.

format

is the format: F is fixed-length, V is variable-length, DIR is a directory. A dash indicates an erased or revoked alias.

lrecl

is the logical record length of the largest record in the file. For directories and revoked or erased aliases, a dash is displayed.

norecs

is the number of logical records in the file. For directories and revoked or erased aliases, a dash is displayed.

noblks

is the number of CMS data blocks that the file occupies. For directories and revoked or erased aliases, a dash is displayed.

mm/dd/yy

is the date (month/day/year) that the file was last updated. For a directory, the date the directory was created is displayed. A dash appears in this column for erased or revoked aliases.

hh:mm:ss

is the time (hours:minutes :seconds) that the file was last updated. For a directory, the time the directory was created is displayed. A dash appears in this column for erased or revoked aliases.

valid

is the volume serial number of the minidisk on which the file resides. For files and subdirectories in SFS directories a dash is displayed.

One entry is displayed for each file or subdirectory listed.

If the SHARE option is specified, the information displayed is:

FILENAME	FILETYPE	FM	OWNER	TYPE	R	W
fn	ft	fm	owner	type	r	w
.
.
.

where:

owner

is the user ID of base file owner. For files on a disk, this is the user ID of the disk owner.

type

is one of the following:

- MDISK for a file on a minidisk
- BASE for a base file in a directory
- DIR for a directory
- ALIAS for an alias in a directory
- ERASED for an erased alias
- REVOKED for a revoked alias

r

is read authority.

X indicates that you have read authority.

— A dash indicates that you do not have read authority.

P means that the authority is managed by an External Security Manager (ESM).

For files on disk, an X will be displayed if the disk is accessed read/only or read/write.

w

is write authority.

X indicates that you have write authority.

— A dash indicates that you do not have write authority.

P means that the authority is managed by an External Security Manager (ESM).

For files on disk, an X will be displayed if the disk is accessed read/write, a dash if accessed read/only.

One entry is displayed for each file or subdirectory listed.

If the SEARCH option is specified, the information displayed is:

FILENAME	FILETYPE	FM	DIRECTORY
fn	ft	fm	directory
.	.	.	.
.	.	.	.
.	.	.	.

where:

directory

is the complete name of directory that contains the file.

One entry is displayed for each file listed. If the specified files are on a disk, you will receive the message:

DMSWFL1182E The SEARCH option may not be used with a minidisk

Messages and Return Codes

DMSLSS002E File *fn ft fm* not found [RC = 28]
 DMSLSS105S Error *nn* writing file *fn ft fm* to XEDIT [RC = 100]
 DMSLSS109E Virtual storage capacity exceeded [RC = 104]
 DMSLSS689E File must be F-format *lrecl* or V-format [RC = 24]
 DMSLSS1160E Directory *dirname* already open. [RC = 70]
 DMSLSS1184E Directory *dirname* not found or you are not authorized for it
 [RC = 28]
 DMSLST003E Invalid option: *option* [RC = 24]
 DMSLST037E Filemode *mode* is accessed as read/only [RC = 36]
 DMSLST048E Invalid mode *mode* [RC = 24]
 DMSLST066E *option1* and *option2* are conflicting options [RC = 24]
 DMSLST069E Filemode *mode* not accessed [RC = 36]
 DMSLST070E Invalid parameter *parameter* [RC = 24]
 DMSLST688E XEDIT option only valid from XEDIT environment [RC = 24]
 DMSLST1182E The SEARCH option may not be used with a minidisk [RC = 74]
 DMSLST1183E '**' may not be specified for the filemode with the SEARCH
 option [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

LISTIO

Use the LISTIO command in CMS/DOS to display a list of current assignments for system and/or programmer logical units in your virtual machine.

Format

LISTIO	<pre> [SYS PROG SYS xxx A UA ALL] </pre>	<pre> [(options... [])] </pre>
	<pre> Options: [EXEC] [STAT] [APPEND] </pre>	

Operands**SYS**

requests a list of the physical devices assigned to all system logical units.

PROG

requests a list of the physical devices assigned to programmer logical units SYS000 through SYS241.

SYSxxx

requests a display of the physical device assigned to the particular logical unit specified.

A

requests a list of only those logical units that have been assigned to physical devices.

UA

requests a list of only those logical units that have not been assigned to physical devices; that is, that are unassigned.

ALL

requests a list of the physical units assigned to all system and programmer logical units. If no operand is specified, ALL is the default.

Options

The EXEC and APPEND options are mutually exclusive; if both are entered on the command line, the last one entered is in effect.

EXEC

erases the existing \$LISTIO EXEC file, if one exists, and creates a new one.

APPEND

adds new entries to the end of an existing \$LISTIO EXEC file. If no \$LISTIO EXEC file exists, a new one is created.

STAT

lists the status (read-only or read/write) of all disks and directories currently assigned.

Usage Notes

1. Logical units are assigned and unassigned with the ASSGN command. For a list of logical units and valid device types, see the discussion of the ASSGN command.
2. The \$LISTIO EXEC contains one record for each logical unit listed. The format is:

```
&1 &2 SYSxxx device
```

or

```
&1 &2 SYSxxx mode [status]
```

where column 1 is blank.

Responses

Depending on the operands specified, the following is displayed for each unit requested in the LISTIO command:

```
SYSxxx device
```

or

```
SYSxxx mode [status]
```

where device is the device type (READER, PRINTER, PUNCH, TERMINAL, TAPn, IGN, or UA). If the device is a disk or directory the one-character mode letter is displayed. If the STAT option is specified, the status (R/O or R/W) is also displayed.

Messages and Return Codes

DMSERD257T	Internal system error at address <i>address</i> (offset <i>offset</i>)
DMSLLU003E	Invalid option: <i>option</i> [RC = 24]
DMSLLU006E	No read/write A filemode accessed [RC = 36]
DMSLLU070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSLLU099E	CMS/DOS environment not active [RC = 40]
DMSLLU105S	Error <i>nm</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSLLU303E	No SYSXXX satisfies request [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814
Errors in closing a file	219

LKED

Use the LKED command to create a CMS LOADLIB or LOADLIB member.

The LKED command behaves differently depending upon the environment it is issued from. In the following description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. The behavior of the LKED command in both environments is described below to help you plan and develop applications that will run in both environments.

Format

LKED	<pre> <i>fname</i> [(options...)] Options: [NCAL] [LET] [ALIGN2] [NE] [OL] [RENT] [REUS] [REFR] [OVLY] [XCAL] [NAME <i>membername</i>] [LIBE <i>libraryname</i>] [XREF] [TERM] [PRINT] [MAP] [NOTERM] [DISK] [LIST] [NOPRINT] [SIZE { <i>value1</i> <i>value2</i> } { <i>value1</i> ,<i>value2</i> }] [AMODE { 24 31 ANY }] [RMODE { 24 ANY }] </pre>
-------------	---

Operands

fn
 specifies the file name of the object file to be processed. The file must have a file type of TEXT and fixed-length, 80-character records.

Options

If you specify duplicate or conflicting linkage editor options, the linkage editor resolves them according to normal procedures found in the *MVS/XA Linkage Editor and Loader User's Guide*, GC26-4143. If you specify duplicate or conflicting CMS-related options, the last one entered on the command line is in effect. The CMS-related options are: TERM, NOTERM, PRINT, DISK, NOPRINT, NAME, LIBE, AMODE, and RMODE.

NCAL

suppresses the automatic library call function of the linkage editor.

LET

suppresses marking of the load module “not executable” in the event of some linkage editor error condition.

ALIGN2

indicates that boundary alignment specified in the linkage editor input file is to be performed on the basis of 2048-byte boundaries. If this option is omitted, alignment is performed on the basis of 4096-byte boundaries.

NE

marks the load module output as “not to be edited” such that it cannot be processed again by the linkage editor.

OL

marks the load module output “only loadable.”

RENT

marks the load module reenterable.

REUS

marks the load module reusable.

REFR

marks the load module refreshable.

OVLY

processes an overlay structure.

XCAL

allows valid exclusive CALLs in the overlay structure.

NAME *membername*

is the member name to be used for the load module created. The member name specified here overrides the default name, but it cannot override a name specified via the linkage editor NAME control statement.

LIBE *libraryname*

is the file name of a LOADLIB file where the output load module is to be placed. The LOADLIB file specified here may also be used for auxiliary input to the linkage editor via the INCLUDE statement.

XREF

produces an external symbol cross-reference for the modules being processed.

MAP

produces only a module map for the processed module(s).

LIST

includes only linkage editor control messages in the printed output file.

TERM

displays any linkage editor diagnostic messages at the user terminal.

NOTERM

suppresses the displaying of diagnostic messages.

PRINT

spools the linkage editor printed output file to the printer.

DISK

stores the linkage editor output in a CMS file with a file type of LKEDIT.

NOPRINT

produces no output file.

SIZE *value1 value2*

indicates the amount of virtual storage to be used by the linkage editor and specifies the portion of that storage to be reserved for the load module buffer. The SIZE parameters must lie within the following limits:

value1 64K to 9999K (or 65536 to 999999)
value2 6K to 100K (or 6144 to 102400)

If either of the SIZE parameters is omitted, the default values are 384K for value1 and 96K for value2. If either value is invalid, the OS Linkage Editor defaults to the 9999K for value1 and 100K for value2, which are the maximum values. Values greater than 999999 can be entered in the form nnnnK (with K equal to 1024). For example, enter 2000K instead of 2048000. Values accepted by the linkage editor are displayed in the output file.

AMODE

specifies the addressing mode in which the program will be entered in a 370-XA mode virtual machine. In a System/370 mode virtual machine, you may specify AMODE, although only 24-bit addressing is available. This allows you to create XA capable module files on a System/370 mode virtual machine. The AMODE defined by this option is placed in the header record of the MODULE file being created.

This option overrides the AMODE determined by the LOAD process. If you specify RMODE, but do **not** specify AMODE, the AMODE for the MODULE is determined from the following default criteria:

- If you specify RMODE ANY, the AMODE specification defaults to AMODE 31.
- If you specify RMODE 24, the AMODE defaults to the AMODE of the entry point determined by the linkage editor.

If you specify neither AMODE nor RMODE, the AMODE for the module is determined by the linkage editor. The valid AMODE values and their meanings are:

24	This entry point is to receive control in 24-bit addressing mode.
31	This entry point is to receive control in 31-bit addressing mode when running in an 370-XA mode virtual machine. Since 31-bit addressing is unavailable in a System/370 mode virtual machine, the entry point will receive control in 24-bit addressing mode when running in a System/370 mode virtual machine.
ANY	This entry point is capable of operating in either 24-bit or 31-bit addressing mode. It will receive control in the addressing mode of its caller.

RMODE

specifies, in an 370-XA mode virtual machine, the location in virtual machine storage where the loaded MODULE is to reside.

In a System/370 mode virtual machine, you may specify RMODE, although only 16Mb of storage is available. This allows you to create XA capable MODULE files on a System/370 mode virtual machine. The RMODE defined by this option overrides the RMODE determined by the linkage editor.

If you specify neither AMODE nor RMODE, the RMODE for the MODULE is determined by the most restrictive RMODE encountered by the linkage editor process.

Note: An AMODE value specified without an RMODE option defaults to RMODE 24 for the module.

If specified, RMODE has the following meaning:

- 24 The load must reside below the 16Mb line in an 370-XA mode virtual machine, overriding the RMODE determined by the linkage editor. An RMODE 24 definition would be reflected in the MODULE header record. In a System/370 mode virtual machine, the load can only reside below 16Mb.
- ANY The load may reside above or below the 16Mb line in an 370-XA mode virtual machine, overriding the RMODE determined by the linkage editor. An RMODE ANY definition would be reflected in the MODULE header record. In a System/370 mode virtual machine, the load can only reside below 16Mb.

Usage Notes

1. Only a subset of the possible linkage editor control statements are meaningful in CMS. Since the CMS interface program cannot examine the input data for the LKED command, all of the control statements are allowed, even though several of them result in the creation of a load module file that cannot be used under CMS. For both command options and control statements, see the publication *MVS/XA Linkage Editor and Loader User's Guide*, GC26-4143.

2. When you use the linkage editor INCLUDE control statement to include a load module, the DDNAME referring to the module library must be other than SYSLMOD and it must have been previously defined by a FILEDEF. If you include a member of the LOADLIB that receives linkage editor output, you can enter statements in the following form:

```
filedef libdef disk mylib loadlib a (recfm u
lked fn (libe mylib)
```

Contents of file FNAME TEXT:

```
include libdef (libmem1)
name libmem2
```

3. The LKED command produces one temporary file:

```
fn SYSUT1
```

This file is temporarily created for each link-edit step; any existing file with the same file identifier is erased at the beginning of the link edit. This file is placed on the read/write disk or directory with the most available space. Work space is automatically allocated as needed during the link edit and returned to available status when the link edit is complete. Insufficient space causes abnormal termination of the link edit.

4. The LKED command produces two permanent files:

```
fn LOADLIB
fn LKEDIT
```

The 'fn LOADLIB' file contains the load module(s) that the linkage editor created. This file is in CMS simulated partitioned data set format, as created by the CMS OS data management macros. The file name of the input file becomes the file name of the LOADLIB file, unless the LIBE option is specified. The file name of the input file also becomes the member name of the output load module, unless either the NAME option or a NAME control statement is used. One or more load modules may be created during a single LKED command

execution if the NAME linkage editor control statement is used in the input file. When the NAME control statement is used, that name becomes the member name in the LOADLIB file. The replace option of the NAME statement determines whether existing members with the same name are replaced or retained.

The 'fn LKEDIT' file contains the printed output listing produced according to the XREF, MAP, or LIST options. This file is created unless the PRINT or NOPRINT option is specified. The LOADLIB and LKEDIT files are placed on (1) the disk or directory from which the input file was read, (2) the parent, or (3) the disk or directory accessed as A. Failure to obtain sufficient space for these files results in abnormal termination of the linkage editor. The specified LOADLIB must not exist on any read/only extension of the disk or directory accessed as A.

5. Because the LKED command uses the OS Linkage Editor for the actual link of the TEXT file to the LOADLIB as an executable module, you can override the file definitions with the CMS FILEDEF command prior to using the LKED command. Refer to the publication *OS/VS Linkage Editor and Loader*, GC26-3813, for information regarding the file definitions.

The default FILEDEF commands issued by the LKED command for the dnames presented to the Linkage Editor are:

```
FILEDEF SYSLIN DISK fn TEXT * (RECFM F BLOCK 80 NOCHANGE
FILEDEF SYSLMOD DISK fn LOADLIB A1 (RECFM U BLOCK 260 NOCHANGE MEMBER mname
```

—or—

```
FILEDEF SYSLMOD DISK libname LOADLIB A1 (RECFM U BLOCK 260 NOCHANGE MEMBER mname
FILEDEF SYSUT1 DISK fn SYSUT1 * (NOCHANGE
FILEDEF SYSPRINT DISK fn LKEDIT A1
```

—or—

```
FILEDEF SYSPRINT PRINTER
```

—or—

```
FILEDEF SYSPRINT DUMMY
```

Note: Notice that the SYSPRINT FILEDEFs are specified without the NOCHANGE option. This way the filedefs are not overridden.

Also note that *mname* may mean either *fn* or *membername*. If you specify the NAME option, *mname* means the file name. If you do not specify the NAME option, *mname* means member name.

6. At the completion of the LKED command, all FILEDEFs that do not have the PERM option are erased.
7. The blocksize of the output LOADLIB is limited to 13K.
8. For information on listing, copying, and compressing a CMS LOADLIB, refer to the LOADLIB command.

Messages and Return Codes

DMSLKD001E	No filename specified [RC = 24]
DMSLKD002E	File <i>fn</i> [<i>ft</i> [<i>fm</i>]] not found [RC = 28]
DMSLKD004W	Warning messages issued [RC = 4]
DMSLKD005E	No <i>option</i> specified [RC = 24]
DMSLKD006E	No read/write filemode accessed [RC = 36]
DMSLKD007E	File <i>fn ft fm</i> [<i>is</i>] not fixed, 80-character records [RC = 32]

DMSLKD008W Error messages issued [RC=8]
DMSLKD012W Severe error messages issued [RC=12]
DMSLKD016W Terminal error messages issued [RC=16]
DMSLKD070E Invalid parameter *parameter* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in erasing a file	145

LOAD

Use the LOAD command to read one or more TEXT files (containing relocatable object code) from a minidisk or directory and to load them into virtual storage, establishing the proper linkages between the files. After the text files are loaded, you can:

- use them as input to GENMOD to create MODULE files
- execute them using the START command.

The LOAD command behaves differently depending upon the environment it is issued from. In the following description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. The behavior of the LOAD command in both environments is described below to help you plan and develop applications that will run in both environments.

Note: The SET LOADAREA command affects the outcome of the LOAD and INCLUDE commands.

Format

LOAD	<p><i>fn ...</i> [(options... [])]</p> <p>Options:</p> <table style="width: 100%; border: none;"> <tr> <td style="border: 1px solid black; padding: 2px;">CLEAR NOCLEAR</td> <td style="border: 1px solid black; padding: 2px;">RESET { <i>entry</i> } * }</td> <td style="border: 1px solid black; padding: 2px;">MAP NOMAP</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">TYPE NOTYPE</td> <td style="border: 1px solid black; padding: 2px;">INV NOINV</td> <td style="border: 1px solid black; padding: 2px;">REP NOREP</td> <td style="border: 1px solid black; padding: 2px;">AUTO NOAUTO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">LIBE NOLIBE</td> <td style="border: 1px solid black; padding: 2px;">START</td> <td style="border: 1px solid black; padding: 2px;">DUP NODUP</td> <td style="border: 1px solid black; padding: 2px;">NORLDSave RLDSave</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">NOPRES PRES</td> <td style="border: 1px solid black; padding: 2px;">HIST NOHIST</td> <td colspan="2"></td> </tr> <tr> <td colspan="2" style="border: 1px solid black; padding: 2px;">RMODE { 24 ANY }</td> <td colspan="2" style="border: 1px solid black; padding: 2px;">AMODE { 24 31 ANY }</td> </tr> <tr> <td colspan="2" style="border: 1px solid black; padding: 2px;">ORIGIN { <i>hexloc</i> } TRANS</td> <td colspan="2"></td> </tr> </table>	CLEAR NOCLEAR	RESET { <i>entry</i> } * }	MAP NOMAP	TYPE NOTYPE	INV NOINV	REP NOREP	AUTO NOAUTO	LIBE NOLIBE	START	DUP NODUP	NORLDSave RLDSave	NOPRES PRES	HIST NOHIST			RMODE { 24 ANY }		AMODE { 24 31 ANY }		ORIGIN { <i>hexloc</i> } TRANS			
CLEAR NOCLEAR	RESET { <i>entry</i> } * }	MAP NOMAP																						
TYPE NOTYPE	INV NOINV	REP NOREP	AUTO NOAUTO																					
LIBE NOLIBE	START	DUP NODUP	NORLDSave RLDSave																					
NOPRES PRES	HIST NOHIST																							
RMODE { 24 ANY }		AMODE { 24 31 ANY }																						
ORIGIN { <i>hexloc</i> } TRANS																								

Operands

fn...

specifies the names of the files to be loaded into storage. The files must have a file type of TEXT and consist of relocatable object code such as that produced by the OS language processors. If a GLOBAL TXTLIB command has been issued, *fn* may indicate the name of a TXTLIB member.

Options

If conflicting options are specified, the last one entered is in effect. Options may be overridden or added when you use the INCLUDE command to load additional TEXT files.

CLEAR

clears the load area in storage before the object files are loaded. Whole page frames are released; the remainder of storage that is not on a page boundary is set to binary zeros.

NOCLEAR

does not clear the load area before loading. NOCLEAR is the default.

RESET *entry*

RESET *

sets the starting location for the programs currently loaded. The operand, entry, must be an external name (for example, CSECT control section or ENTRY) in the loaded programs. If RESET is not specified, the default entry point is used. (See Usage Note 4.) If * is entered the results are the same as if the RESET option were omitted.

Note: You should not use the RESET option when loading TEXT files created by any of the following OS/VS language processors under CMS:

- OS Code and Go FORTRAN
- OS FORTRAN IV (G1)
- OS FORTRAN IV (H) Extended
- OS/VS COBOL Compiler and Library
- OS Full American National Standard COBOL Version 4 Compiler and Library.

MAP

writes a load map on your minidisk or directory accessed as A, named LOAD MAP A5. MAP is the default.

NOMAP

does not create the LOAD MAP file.

TYPE

displays the load map at your terminal and writes it on the minidisk or directory accessed as A.

NOTYPE

does not display the load map at the terminal. NOTYPE is the default.

INV

includes invalid card images in the load map. INV is the default.

NOINV

does not include invalid card images in the load map.

REP

includes Replace (REP) statements in the load map. REP is the default.

NOREP

does not include the Replace (REP) statements in the load map.

AUTO

searches your minidisks and directories for TEXT files to resolve undefined references. AUTO is the default.

LOAD

NOAUTO

suppresses automatic searching for TEXT files.

LIBE

searches the text libraries for missing subroutines. If text libraries are to be searched for TEXT files, they must previously have been defined by a GLOBAL command. LIBE is the default.

NOLIBE

does not search the text libraries for unresolved references.

START

executes the program being loaded when loading is completed. LOAD does not normally begin execution of the loaded files. To begin execution immediately upon successful completion of loading, specify START. Execution begins at the default entry point. (See Usage Note 3.)

DUP

displays warning messages at your terminal when a CSECT having the same name as another CSECT that has already been loaded is encountered during processing. The CSECT with the duplicate name is not loaded. DUP is the default. (See Usage Note 2.)

NODUP

does not display warning messages at your terminal when CSECTs having the same name (duplicate CSECTs) are encountered during processing. Only the first CSECT by a given name is loaded, the CSECT with the duplicate name is not loaded.

NORLDSav

instructs the CMS LOADER not to save the relocation information from the TEXT files. NORLDSav is the default.

RLDsave

instructs the CMS LOADER to save the relocation information from the text files. The GENMOD command uses relocation information to generate relocatable CMS module files.

NOPRES

instructs CMS to release the storage that the currently loaded set of programs occupies. These programs can no longer be referenced by another program. NOPRES is the default.

PRES

instructs CMS not to release the storage that the currently loaded set of programs occupies. These programs can be accessed using hard coded addresses but cannot be executed with a START command. The loader tables are rewritten.

HIST

saves the history information (comments) from text files. This information is later included in the module generated by a GENMOD command. The history information requested from text files specified in subsequent INCLUDE commands is also included in the module when you issue the GENMOD command. Only history information from the last LOAD command and its subsequent INCLUDE commands is included in the module by the GENMOD command.

NOHIST

does not save history information from text files. NOHIST is the default.

AMODE

specifies the addressing mode in which the program will be entered in an 370-XA mode virtual machine. In a System/370 mode virtual machine, you may specify AMODE, although only 24-bit addressing is available. This allows you to create XA capable module files on a System/370 mode virtual machine. The AMODE defined by this option propagates to the GENMOD process.

This option overrides the AMODE, reflected in the TEXT ESD record, that is specified at assembly time. If you specify RMODE/ORIGIN, but do **not** specify AMODE, the AMODE for the MODULE is determined from the following default criteria:

- If you specify RMODE ANY, the AMODE specification defaults to AMODE 31.
- If you specify ORIGIN yyyyyyyy, and yyyyyyyy is an address above 16 Mb, the AMODE defaults to AMODE 31.
- The AMODE defaults to the AMODE of the entry point on the ESD record if you specify:
 - RMODE 24
 - ORIGIN xxxxxxxx, and xxxxxxxx is an address below 16 Mb
 - ORIGIN TRANS

If you specify neither AMODE nor RMODE, the AMODE is determined by the AMODE defined in the TEXT ESD for the entry point. The valid AMODE values and their meanings are:

- | | |
|-----|---|
| 24 | This entry point is to receive control in 24-bit addressing mode. |
| 31 | This entry point is to receive control in 31-bit addressing mode when running in an 370-XA mode virtual machine. Since 31-bit addressing is unavailable in a System/370 mode virtual machine, the entry point will receive control in 24-bit addressing mode when running in a System/370 mode virtual machine. |
| ANY | This entry point is capable of operating in either 24-bit or 31-bit addressing mode. It will receive control in the addressing mode of its caller when control is passed to the entry point. |

RMODE

specifies, in an 370-XA mode virtual machine with greater than 16Mb of storage, the location where the loaded program(s) is to reside.

In a System/370 mode virtual machine, you may specify RMODE, although only 16Mb of storage is available. This allows you to create XA capable MODULE files on a System/370 mode virtual machine. The RMODE defined by this option propagates to the GENMOD process.

This option is mutually exclusive with the ORIGIN option and overrides the RMODE, reflected in the TEXT(s) ESD record, that is specified at assembly time.

If you specify ORIGIN, the RMODE is determined by the ORIGIN definition.

If you specify neither RMODE nor ORIGIN, the RMODE for the program is determined from the most restrictive RMODE encountered in TEXT ESD processing for this load.

Note: An AMODE value specified without an RMODE option defaults to RMODE 24 for the module.

LOAD

If specified, RMODE will cause the file to load above or below the 16Mb line of a 370-XA mode virtual machine. The AMODE defined by this option is placed in the header record of the or below the 16Mb line, starting at the beginning of the largest contiguous area available.

If you specify neither AMODE nor RMODE, the RMODE is determined by the RMODE defined in the TEXT ESD for the entry point. The valid RMODE values and their meanings are:

24 The load resides below the 16Mb line in an 370-XA mode virtual machine, overriding the RMODE definitions encountered on the TEXT ESD records during this load. In a System/370 mode virtual machine, the load can only reside below 16Mb. An RMODE 24 definition is propagated to the GENMOD process.

ANY The load resides above the 16Mb line in an 370-XA mode virtual machine, overriding the RMODE definitions encountered on the TEXT ESD records during this load. In a System/370 mode virtual machine, the load can only reside below 16Mb. An RMODE ANY definition is propagated to the GENMOD process.

ORIGIN *hexloc*

ORIGIN TRANS

specifies where CMS loads the program. This location must be in the CMS transient area or in any free CMS storage.

ORIGIN *hexloc* loads the program at *hexloc*. If *hexloc* is within the transient area, the name is not updated. The *hexloc* is a hexadecimal number of up to eight characters.

ORIGIN TRANS loads the program into the CMS nucleus transient area.

The RMODE that results from this option propagates to the start of the GENMOD process:

- ORIGIN xxxxxxxx

If xxxxxxxx is below 16Mb, the result is RMODE 24.

- ORIGIN yyyyyyyy

If yyyyyyyy is above 16Mb, the result is RMODE ANY.

- ORIGIN TRANS

The result is RMODE 24.

IMPORTANT

If you do not specify ORIGIN/RMODE, loading begins at the storage area defined by the SET LOADAREA command.

You may issue the LOAD command with various ORIGIN/RMODE and/or AMODE options:

LOAD pgmA pgmB ...

LOAD pgmA pgmB ...(RMODE xxx AMODE xxx

LOAD pgmA pgmB ...(RMODE xxx

LOAD pgmA pgmB ...(ORIGIN xxx AMODE xxx

LOAD pgmA pgmB ...(ORIGIN xxxx

LOAD pgmA pgmB ...(AMODE xxx

The chart in Figure 8 shows how RMODE and AMODE are determined, and where the loaded programs will reside in storage at the end of the LOAD process.

- “LOAD CSECT” relates to the first program specified on the LOAD command.
- “SUBSEQUENT CSECT” is any other program specified on the LOAD command or referenced by any program specified on the LOAD command and/or INCLUDE commands.

LOAD

LOAD command options		On TEXT ESD record				RMODE/ AMODE at end of load process	Residency of loaded programs	
ORIGIN/ RMODE	AMODE	LOAD CSECT		SUBSEQUENT CSECT			370 virt mach	XA * virt mach
		RMODE	AMODE	RMODE	AMODE			
▶ 16Mb/ ANY	24	n/a	n/a	n/a	n/a	993E 377E	993E 377E	993E 377E
▶ 16Mb/ ANY	31	n/a	n/a	n/a	n/a	ANY/31	◀16Mb	▶16Mb
▶ 16Mb/ ANY	ANY	n/a	n/a	n/a	n/a	ANY/31 *ANY/ANY	n/a ◀16Mb	▶16Mb ◀16Mb
◀ 16Mb/ 24	24	n/a	n/a	n/a	n/a	24/24	◀16Mb	◀16Mb
◀ 16Mb/ 24	31	n/a	n/a	n/a	n/a	24/31	◀16Mb	◀16Mb
◀ 16Mb/ 24	ANY	n/a	n/a	n/a	n/a	24/ANY	◀16Mb	◀16Mb
▶ 16Mb/ ANY		n/a	n/a	n/a	n/a	ANY/31	◀16Mb	▶16Mb
◀ 16Mb/ 24		n/a	24	n/a	n/a	24/24	◀16Mb	◀16Mb
◀ 16Mb/ 24		n/a	31	n/a	n/a	24/31	◀16Mb	◀16Mb
◀ 16Mb/ 24		n/a	ANY	n/a	n/a	24/ANY	◀16Mb	◀16Mb
◀ 16Mb/ 24		n/a	n/a	n/a	*	24/*	◀16Mb	◀16Mb
	24	n/a	n/a	n/a	n/a	24/24	◀16Mb	◀16Mb
	31	n/a	n/a	n/a	n/a	24/31	◀16Mb	◀16Mb
	ANY	n/a	n/a	n/a	n/a	24/ANY	◀16Mb	◀16Mb
		*	*	n/a	n/a	24/24	◀16Mb	◀16Mb
		24	24	n/a	n/a	24/24	◀16Mb	◀16Mb
		24	31	n/a	n/a	24/31	◀16Mb	◀16Mb
		24	ANY	n/a	n/a	24/ANY	◀16Mb	◀16Mb
		ANY	31	24	n/a	994W 24/31	994W ◀16Mb	994W ◀16Mb
		ANY	31	24	*	994W 24/*	994W ◀16Mb	994W ◀16Mb
		ANY	31	ANY	n/a	ANY/31	◀16Mb	▶16Mb
		ANY	ANY	ANY	n/a	ANY/31 *ANY/ANY	n/a ◀16Mb	▶16Mb ◀16Mb
		ANY	n/a	ANY	*	ANY/31	◀16Mb	▶16Mb

Figure 8. LOAD Process, RMODE and AMODE Determination, Load Residency

Note: Use the numbers on the side of the chart to determine the meaning of the asterisk (*) in each column.

1. If a 370-XA mode virtual machine has less than 16Mb of storage, the loaded programs can only reside below 16Mb. A specification of

RMODE ANY, either on the LOAD command or TEXT ESD, would result in the LOAD residing below the 16Mb line.

2. If a 370-XA mode virtual machine has less than 16Mb of storage, the loaded programs can only reside below 16Mb. In this case, they may execute in either 24 or 31-bit addressing mode. If you specified AMODE ANY on the LOAD command or defaulted to a TEXT ESD having an AMODE ANY defined for the entry point, AMODE ANY would result at the end of the load process.
3. If you specified an entry point using the RESET option, the AMODE at the end of the load process would reflect the AMODE defined on the TEXT ESD for that entry point.
4. If no RMODE and AMODE is specified (i.e. blanks), the defaulted RMODE and AMODE is 24.
5. If you specified an entry point using the RESET option on the LOAD or INCLUDE command, and the load is above the 16Mb line, the only valid AMODE is 31. Assembler H version 2 does not allow a combination of RMODE ANY and AMODE 24. An assembly error would occur.

Usage Notes

1. Unless the NOMAP option is specified, you must have a read/write minidisk or directory accessed as A when you issue the LOAD command. The loader creates a load map each time the LOAD command is issued without the NOMAP option. A load map is a file that contains the location of control sections and entry points of files loaded into storage. This load map is named LOAD MAP A5. Each time LOAD is issued, a new LOAD MAP file replaces any previous LOAD MAP file.

LOAD MAP information is provided by the MAP option and contains the AMODE and RMODE information specified on the TEXT ESD record for the loaded program. You can use the information to detect programs with AMODE or RMODE values different from those specified on the LOAD command. If you specify the TYPE option, you can read the information in the LOAD MAP file and on your terminal. It appears in the following format:

```

PROG1 SD 00020000          RMODE 24  AMODE 24
PROG2 SD 00020318          RMODE ANY AMODE 31
PROG3 SD 00020B48          RMODE 24  AMODE ANY

```

If invalid card images exist in the file or files that are being loaded, they are listed with the message INVALID CARD in the LOAD MAP file. To suppress this listing in the load map, use the NOINV option.

If Replace (REP) statements exist in the file being loaded, they are included in the LOAD MAP file. To suppress this listing of REP statements, specify the NOREP option.

If the ENTRY or LIBRARY control cards are encountered in the file, the load map contains an entry:

```
CONTROL CARD- ...
```

listing the card that was read.

Mapping of any common areas that exist in the loaded files will occur when the program is prepared for execution by the START or GENMOD command or by

the **START** option of the **LOAD** or **INCLUDE** command. An updated load map may be displayed before program execution if the **START** command is issued with the **NO** option to suppress execution.

2. CSECTs (control sections) having a duplicate name (duplicate CSECTs) are bypassed by the loader. Only the first CSECT encountered is physically loaded. The CSECTs with the duplicate names are not loaded. A warning message is displayed at your terminal if you specified the **DUP** option. If a section contains an **ADCON** that references a CSECT with a duplicate name that has not been loaded, that **ADCON** may be resolved incorrectly.
3. The loader selects the entry point for the loaded program according to the following hierarchy:
 - From the parameter list on the **START** command
 - From the last **RESET** operand in a **LOAD** or **INCLUDE** command
 - From the last **ENTRY** statement in the input
 - From the last **LDT** statement in the input
 - From the first assembler- or compiler-produced **END** statement that specifies an entry point if no **ENTRY** statement is in the input
 - From the first byte of the first control section of the loaded program if there is no **ENTRY** statement and no assembler- or compiler-produced **END** statement specifying an entry point
4. The **LOAD** command should not be used to execute programs containing **DOS** macros. To link-edit and execute programs in the **CMS/DOS** environment, use the **DOSLKED** and **FETCH** commands.
5. When the text to be loaded contains dummy sections or defines pseudo registers, their cumulative length must not exceed 32767 (**X'7FFF'**). If this limit is exceeded, the values stored by the **CXD** entry will not be accurate and the load map will not contain the correct cumulative length values.
6. Common sections and pseudo-register cumulative length files (**CXD**s) are not resolved until either a **GENMOD** command or **START** command is issued (or the **START** option of the **LOAD** or **INCLUDE** command). Be sure to resolve everything before performing functions such as a **CP SAVESYS**.
7. See Figure 9 on page 317 for an illustration of the loader search order. The loader uses this search order to locate the file name on the **LOAD** and **INCLUDE** command lines, as well as in the handling of unresolved references.
8. When you specify the **HIST** option, up to 819 comment records can be saved from text files. These records are included later in the module generated by a **GENMOD** command. When this limit is exceeded, a message is displayed and a warning is placed in the last record of the history data to indicate this condition. Any history information exceeding 819 records is not included.

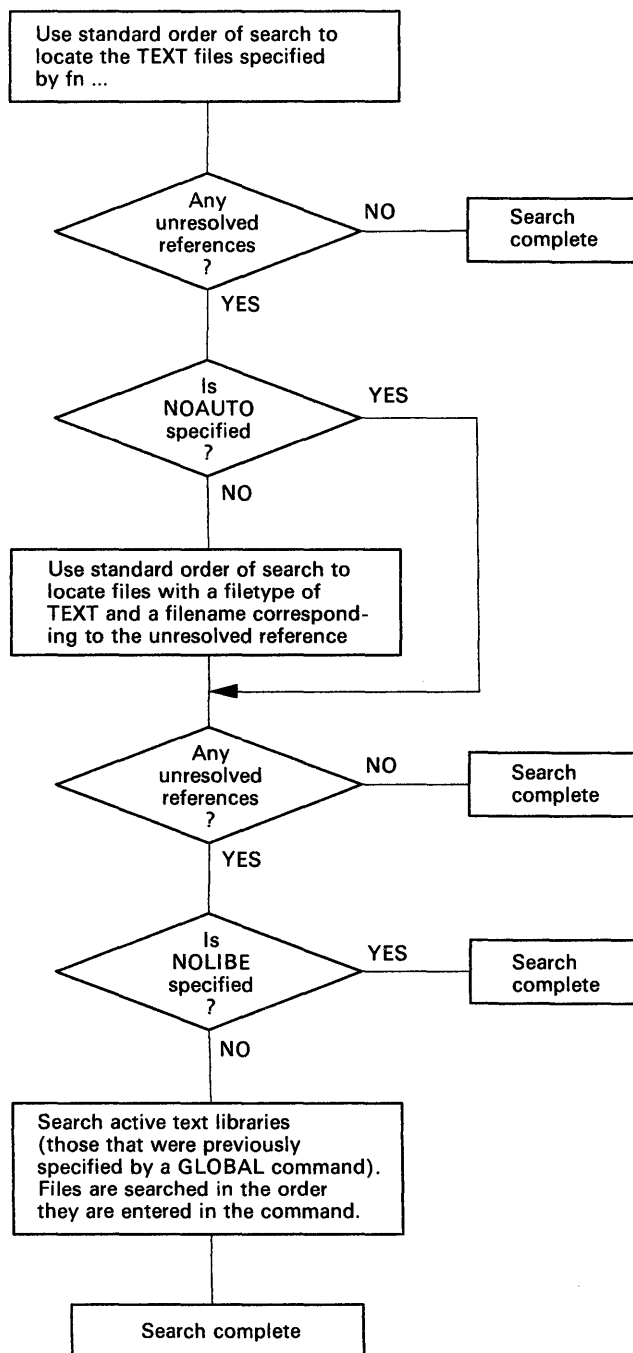


Figure 9. Loader Search Order

9. The CMS loader also loads routines called dynamically by OS LINK, LOAD, and XCTL macros. Under certain circumstances, an incorrect entry point may be returned to the calling program.

10. If you specify neither ORIGIN nor RMODE on the LOAD command, and if you did not issue SET LOADAREA 20000 (i.e. no overrides are requested), the LOAD will start loading above or below the 16Mb line based on the RMODE definition of the first TEXT ESD record encountered.

LOAD

- If a more restrictive RMODE is encountered, RMODE 24 after RMODE ANY was the first detected, the LOAD will stop loading above 16Mb and will start loading below that line. If this occurs, you will receive a message.
 - LOAD process will continue, unless no more storage is available to contain the load.
11. LOAD does not clear user storage unless the CLEAR option is specified.
 12. When you specify the RLDSAVE option, the CMS LOADER can save the relocation information:
 - A load created below the 16Mb line can save up to 16,383 address constants.
 - A load created above the 16Mb line can save up to 26,214 address constants.
 13. Issuing a LOAD command with the RLDSAVE option will result in a relocatable module, even if the RLDSAVE option is not specified on a subsequent INCLUDE command. All RLD data associated with these programs will be saved.
 14. If you issue a LOAD command with the NORLDSAV option, and specify RLDSAVE on a subsequent INCLUDE command, the RLD data will be saved from the time you issued the INCLUDE command. The module will be labeled relocatable, but may not function correctly.
 15. If you specify your first CSECT with a length of zero(0), the load map address will be specified as asterisks. These asterisks will be at the same location as the first valid address specified on the map after the LOAD has completed. Determine the actual address in one of the following ways:
 - Use the address produced by the map.
 - Use the PROGMAP command.

Loader Control Statements

You can add loader control statements to TEXT files either by editing them or by punching real cards and adding them to a punched text deck before reading it into your virtual machine. The seven control cards recognized by the CMS loader are discussed below.

The ENTRY and LIBRARY cards, which are discussed first, are similar to the OS linkage editor control statements ENTRY and LIBRARY. The CMS ENTRY and LIBRARY statements must be entered beginning in column 1.

ENTRY Statement: The ENTRY statement specifies the first instruction to be executed. It can be placed before, between, or after object modules or other control statements. The format of the ENTRY statement is shown in Table 8. The external name is the name of a control section or an entry name in the input deck. It must be the name of an instruction, not of data.

ENTRY	external name
-------	---------------

LIBRARY Statement: The LIBRARY statement can be used to specify the never-call function. The never-call function (indicated by an asterisk (*) as the first operand) specifies those external references that are not to be resolved by the

automatic library call during any loader step. It is negated when a deck containing the external name referred to is included as part of the input to the loader. The format of the LIBRARY statement is shown in Table 9. The external reference refers to an external reference that may be unresolved after input processing. It is not to be resolved. Multiple external references within the parentheses must be separated by commas. The LIBRARY statement can be placed before, between, or after object decks or other control statements.

LIBRARY	* (external reference)

Loader Terminate (LDT) Statement: The LDT statement is used in a text library as the last record of a member. It indicates to the loader that all records for that member were processed. The LDT statement can contain a name to be used as the entry point for the loaded member. The LDT statement has the format shown in Table 10.

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.
2-4	LDT — identifies type of statement.
5-16	Not used.
17-24	Blank or entry name (left-justified and padded with blanks to eight characters).
25	Blank.
26-33	May contain information specified on a SETSSI card processed by the TXTLIB command.
34-72	May be used for comments or left blank.
73-80	Not used by the loader. You may leave these columns blank or insert program identification for your own convenience.

Include Control Section (ICS) Statement: The ICS statement changes the length of a specified control section or defines a new control section. It should be used only when REP statements cause a control section to be increased in length.

The format of an ICS statement is shown in Table 11. An ICS statement must be placed at the front of the file or TEXT file.

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.

Table 11 (Page 2 of 2). ICS Statement Format	
Column	Contents
2-4	ICS — identifies the type of load statement.
5-15	Blank.
16	, (comma).
17-24	Control section name — left-justified in these columns.
25-28	Hexadecimal length in bytes of the control section. This length must be greater than zero, and must not be less than the actual length of a previously specified control section. It must be right-justified in columns with unused leading columns filled with blanks or zeros.
29-72	May be used for comments or left blank.
73-80	Not used by the loader. You may leave these columns blank or insert program identification for your own convenience.

Note: Only six characters can be coded for the CSECT name in the ICS statement, but the loader compares eight characters to the CSECT name from the TEXT file.

Set Location Counter (SLC) Statement: The SLC statement sets the location counter used with the loader. The file loaded after the SLC statement is placed in virtual storage beginning at the address set by this SLC statement.

The SLC statement has the format shown in Table 12. It sets the location counter in one of three ways:

1. With the absolute virtual address specified as a hexadecimal number in columns 5-12.
2. With the symbolic address already defined as a program name or entry point. This is specified by a symbolic name punched in columns 17-22.
3. If both a hexadecimal address and a symbolic name are specified, the absolute virtual address is converted to binary and added to the address assigned to the symbolic name; the resulting sum is the address to which the loader's location counter is set. For example, if 000000F8 was specified in columns 5-12 of the SLC card image and GAMMA was specified in columns 17-22, where GAMMA has an assigned address of 006100 (hexadecimal), the absolute address in columns 5-12 is added to the address assigned to GAMMA giving a total of 0061F8. Thus, the location counter would be set to 0061F8.

Note: Be careful when using absolute values on a 370-XA mode virtual machine with LOADAREA set to RESPECT.

Table 12 (Page 1 of 2). SLC Statement Format	
Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.

Column	Contents
2-4	SLC — identifies the type of load statement.
5-12	Hexadecimal address to be added to the value of the symbol, if any, in columns 17-22. It must be right-justified in these columns, with unused leading columns filled with blanks or zeros.
13-16	Blank.
17-22	Symbolic name whose assigned location is used by the loader. Must be left-justified in these columns. If blank, the address in the absolute field is used.
23	Blank.
24-72	May be used for comments or left blank.
73-80	Not used by the loader. You may leave these columns blank or insert program identification for your own convenience.

Replace (REP) Statement: A REP statement allows instructions and constants to be changed and additions made. The REP statement must be punched in hexadecimal code.

The format of a REP statement is shown in Table 13. The data in columns 17-70 (excluding the commas) replaces what has already been loaded into virtual storage, beginning at the address specified in columns 5-12. REP statements are placed in the file either (1) immediately preceding the last statement (END statement) if the text deck does not contain relocatable data such as address constants, or (2) immediately preceding the first RLD (relocatable dictionary) statement if there is relocatable data in the text deck. If additions made by REP statements increase the length of a control section, an ICS statement, which defines the total length of the control section, must be placed at the front of the deck.

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.
2-4	REP — identifies the type of load statement.
5-12	Hexadecimal starting address of the area to be replaced as assigned by the assembler. It must be right-justified in these columns with unused leading columns filled with blanks or zeros.
13-14	Blank.
15-16	ESID (External Symbol Identification) — the hexadecimal number assigned to the control section in which replacement is to be made. The LISTING file produced by the compiler or assembler indicates this number.

Column	Contents
17-70	A maximum of 11 four-digit hexadecimal fields, separated by commas, each replacing one previously loaded halfword (two bytes). The last field must not be followed by a comma.
71-72	Blank.
73-80	Not used by the loader. This field may be left blank or program identification may be inserted.

Set Page Boundary (SPB) Statement: An SPB statement instructs the loader to update the location counter to point to the next page boundary.

The SPB statement has the format shown in Table 14. This statement can be placed before, between, or after object modules or other control statements.

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.
2-4	SPB — identifies the type of load statement.
5-72	May be used for comments or left blank.
73-80	Not used by the loader. This field may be left blank or program identification may be inserted.

Responses

DMSLI0740I Execution begins ...

START was specified with LOAD and the loaded program starts execution. Any further responses are from the program.

INVALID CARD - xxx...xxx

INV was specified with LOAD and an invalid statement was found. The message and the contents of the invalid statement (xxx...xxx) are listed in the file LOAD MAP. The invalid statement is ignored and loading continues.

Messages and Return Codes

DMSERD257T	Internal system error at address <i>addr</i> (offset <i>offset</i>)
DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC=31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSLGT002I	File <i>fn</i> TXTLIB not found
DMSLIO001E	No filename specified [RC=24]
DMSLIO002E	File(s) <i>fn</i> TXTLIB not found [RC=28]
DMSLIO003E	Invalid option: <i>option</i> [RC=24]
DMSLIO005E	No <i>option</i> specified [RC=24]
DMSLIO021E	Entry point <i>name</i> not found [RC=40]
DMSLIO029E	Invalid parameter <i>mode</i> in the RMODE AMODE option field. [RC=24]
DMSLIO055E	No entry point defined [RC=40]

DMSLIO056E File *fn ft [fm]* contains invalid record formats [RC = 32]
 DMSLIO099E CMS/DOS environment not active [RC = 40]
 DMSLIO104S Error *nn* reading file *fn ft fm* from disk or directory
 [RC = 31|55|70|76|99|100]
 DMSLIO105S Error *nn* writing file *fn ft fm* on disk or directory
 [RC = 31|55|70|76|99|100]
 DMSLIO109S Virtual storage capacity exceeded [RC = 104]
 DMSLIO116S Loader table overflow [RC = 104]
 DMSLIO168S Pseudo register table overflow [RC = 104]
 DMSLIO169S ESDID table overflow [RC = 104]
 DMSLIO201W The following names are undefined: *namelist* [RC = 4]
 DMSLIO202W Duplicate identifier *identifier* [RC = 4]
 DMSLIO203W SET LOCATION COUNTER *name* undefined [RC = 4]
 DMSLIO206W Pseudo register alignment error [RC = 4]
 DMSLIO377E AMODE of 24 specified with RMODE of ANY, LOAD failed.
 [RC = 68]
 DMSLIO379E INCLUDE address at or above 16Mb conflicts with LOAD
 address below 16Mb, INCLUDE failed. [RC = 88]
 DMSLIO379E INCLUDE address below 16Mb conflicts with LOAD address at
 or above 16Mb, INCLUDE failed. [RC = 88]
 DMSLIO380E Storage at origin *addr* in use, *file* not loaded. [RC = 104]
 DMSLIO381E Insufficient storage available below 16Mb to load *file*. [RC = 88]
 DMSLIO623S Module cannot be loaded at location *location*—this area is
 available for system use only [RC = 88]
 DMSLIO625S There are too many items that require relocation to save all of the
 RLD information [RC = 104]
 DMSLIO749W There are too many comments in text files to save all of the
 history information [RC = 4]
 DMSLIO907T I/O error on file *fn ft fm* [RC = 31|55|70|76|99|256]
 DMSLIO943E Invalid AMODE *mode* specified [RC = 24]
 DMSLIO993E AMODE of 24 cannot be specified with ORIGIN address greater
 than 16Mb, LOAD failed. [RC = 68]
 DMSLIO994W Restrictive RMODE encountered in CSECT *cname*. LOAD
 continues below 16Mb.
 DMSLIO997E The specified ORIGIN address is outside the virtual machine size,
 LOAD|INCLUDE failed. [RC = 64]
 DMSLIO1220E ORIGIN is invalid when specified with RMODE. [RC = 68]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

LOADLIB

Use the LOADLIB command to list, copy, or compress a CMS LOADLIB. CMS LOADLIBs can be merged, and specified members can optionally be selected or excluded during the merge.

Format

LOADLIB	<pre> [LIST fileid1 COMPRESS fileid1 COPY fileid1] [fileid2 [fileid3]] [(options...[])] Options: [TERM] [REPLACE] [PRINT] [MODIFY] [DISK] SYSIN Control Statements (COPY function only): [SELECT] [EXCLUDE] </pre>
----------------	---

Operands

LIST

lists by member name, the contents of the CMS LOADLIB specified by *fileid1*, and gives a hexadecimal representation of each member's size. Specifying a *fileid1* that is a concatenated SYSUT1 results in an error message indicating that this is not supported.

COMPRESS

recreates a LOADLIB with the same name as the specified file (*fileid1*), and deletes all obsolete members from the new data set.

COPY

copies members of *fileid1* into *fileid2*. If *fileid2* already exists, MODIFY or REPLACE must be specified. If you specify MODIFY, existing members are not replaced in the output data set, but new members are added. If you specify REPLACE, existing members are replaced in the output data set and new members are added.

You must specify SYSIN control statements. If you do not specify SYSIN control statements in a SYSIN dataset (*fileid3*), you will be prompted for them at the terminal with the message: "ENTER:"

Note: You may specify the LOADLIB function (LIST, COMPRESS, COPY) either on the command line or in the SYSIN data set (*fileid3*). If you specify the function in the SYSIN data set, you must issue the FILEDEF command for *fileid1*, *fileid2* (if required), and *fileid3* before you issue the LOADLIB command. However, if you specify the function on the command line, *fileid1*, and optionally, *fileid2* and *fileid3* may be specified either on the command line or defined via FILEDEF commands. Any FILEDEF commands issued by the user remain in effect after the command function completes. During subsequent use of LOADLIB functions, file definitions

which have not been cleared or reissued may override the file identifiers entered in the LOADLIB command line.

fileid1

is the file name, file type, and file mode of the input LOADLIB. This data set is referred to as the SYSUT1 data set. SYSUT1 is always required. An OS load library may not be specified as input.

fileid2

is the file name, file type, and file mode of the output LOADLIB. This data set is referred to as the SYSUT2 data set. If the SYSUT2 data set already exists, either MODIFY or REPLACE must be specified. If a SYSUT2 data set is not specified, LOADLIB SYSUT2 A (or the file mode of the first available read/write disk or directory) is the default. When the default SYSUT2 file is used and no errors occur, *fileid1* is erased and the new file is renamed *fileid1*. SYSUT2 is ignored for the LIST or COMPRESS functions.

fileid3

is the file name, file type, and file mode of the control data set. This data set is referred to as the SYSIN data set. If no SYSIN data set is specified, the user is prompted at the terminal to enter LOADLIB functions or SYSIN COPY control statements.

Options Entered in the Command Line

TERM

directs printer output to the terminal. TERM is the default.

PRINT

directs printer output to the printer.

DISK

directs printer output to a disk or directory. The DISK option creates a file named LOADLIB LISTING *, where "*" is the file mode of the first available read/write disk or directory.

REPLACE

replaces existing members of a data set and adds new members.

MODIFY

does not replace existing members of a data set; adds new members.

SYSIN Control Statements for the Copy Function

SELECT

copies only selected members of a data set. Each member to be copied must be named in a separate line entry following the SELECT statement. Note that if you specify the SELECT statement, the LOADLIB command does not replace existing members of a data set. If you want to replace an existing member of a data set, you must specify (R) immediately following the member name.

EXCLUDE

copies a whole data set except for a few members. Each member to be excluded must be named in a separate line entry following the EXCLUDE statement. You can exclude up to 256 members for each COPY function.

Note: Indicate the end of control statements from the terminal by entering a null line; EOF serves this purpose in a SYSIN file. If you want to copy an entire data set, specify COPY and enter a null line at the terminal (or include a blank line in a SYSIN file). To avoid unexpected results, clear the file definitions used by the

LOADLIB

COPY function before specifying new file identifiers in subsequent LOADLIB commands.

Usage Notes

1. If a LOADLIB COPY or COMPRESS into an existing LOADLIB results in a CMS ABEND001, check the integrity of the LOADLIB directory. If the file, \$PDSTEMP LOADLIB, exists on your disk or SFS directory, **do not erase it**. The \$PDSTEMP LOADLIB file contains the updated LOADLIB directory. Reissue another LOADLIB COPY or COMPRESS command where the modified output LOADLIB is the SYSUT1 data set and omit the SYSUT2 data set from the command input. If the command is successful, the LOADLIB's directory will be restored.
2. To select or exclude members of a data set, enter the LOADLIB COPY command and press ENTER. When you receive the status message "VM READ" or "Enter a command or press a PF or PA key," type in SELECT or EXCLUDE (or press ENTER to copy the entire data set). When you receive another status message "VM READ" or "Enter a command or press a PF or PA key," type the name of the first data set member to be selected or excluded and press ENTER. For each successive status message of "VM READ" or "Enter a command or press a PF or PA key," type the name of the dataset to be selected or excluded. When you have specified each data set member that you wish to select or exclude, enter a null line to end the command.
3. Although SELECT and EXCLUDE are both valid control statements for the LOADLIB COPY command, they must be used separately. Interchanging SELECT and EXCLUDE statements within the same command routine may lead to unpredictable results.

Responses

MEMBER - member name HAS BEEN COPIED|EXCLUDED
ALIAS - alias name HAS BEEN COPIED|EXCLUDED
ALIAS - alias name NOT COPIED. MEMBER member name FOR THE ALIAS NOT FOUND. An EXCLUDE or SELECT statement controlled the COPY function, and the SYSUT2 data set did not contain the parent member for the alias. The member may not have been moved from the SYSUT1 data set because it was excluded, it was not specified in the SELECT list, or the member name did not precede the alias name in the SELECT list.

MEMBER - member name HAS BEEN REPLACED IN DATA SET
MEMBER - member name DOES NOT EXIST BUT HAS BEEN ADDED TO DATA SET. REPLACE was specified but the member was not in the output data set, therefore the member was added to the output data set.

MEMBER - member name COPY UNSUCCESSFUL
An error occurred while trying to add/replace the member in the output data set. (For example, if MODIFY was specified and the member already existed in the output data set.) The COPY continues with the next member to be copied.

MEMBER - member name NOT FOUND
The member requested was not found in the input data set.

MEMBER - member name NOT COPIED. WRONG LENGTH NOTE LIST FOUND.
MEMBER - member name NOT COPIED. NOTE LIST UPDATE LOGIC ERROR.
USER TTR WAS NOT UPDATED
NOTE LIST TTR OR RECORD WAS NOT UPDATED

Messages and Return Codes

DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC = 31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSUTL003E	Invalid option: <i>option</i> [RC = 24]
DMSUTL014E	Invalid function <i>function</i> [RC = 24]
DMSUTL024E	File <i>fn ft fm</i> already exists [RC = 28]
DMSUTL032E	Invalid filetype <i>ft</i> [RC = 24]
DMSUTL037E	Output filemode <i>mode</i> is accessed as read/only [RC = 36]
DMSUTL039E	No entries in library <i>fn ft fm</i> [RC = 32]
DMSUTL042E	No fileid(s) specified [RC = 24]
DMSUTL047E	No function specified [RC = 24]
DMSUTL054E	Incomplete fileid specified [RC = 24]
DMSUTL065E	<i>option</i> option specified twice [RC = 24]
DMSUTL066E	<i>option1</i> and <i>option2</i> are conflicting options [RC = 24]
DMSUTL069E	Filemode <i>mode</i> is not accessed [RC = 36]
DMSUTL073E	Unable to open file <i>ddname</i> [RC = 28]
DMSUTL188W	SYSUT2 header is invalid because of blocksize incompatibility; user action required [RC = 4]
DMSUTL189E	The LIST function of the LOADLIB command does not support concatenated SYSUT1 [RC = 24]
DMSUTL901T	Unexpected error at <i>vstor1</i> : plist <i>function fn ft fm</i> at <i>vstor2</i> , base <i>vstor3</i> , rc = <i>nn</i> [RC = 31 55 70 76 99 256]
DMSUTL907T	I/O error on file <i>fn ft fm</i> [RC = 256]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814
Errors in copying a file	70

LOADMOD

Use the LOADMOD command to load a CMS module into storage.

The LOADMOD command behaves differently depending upon the environment it is issued from. In the following description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. The behavior of the LOADMOD command in both environments is described below to help you plan and develop applications that will run in both environments.

In an 370-XA mode virtual machine, CMS will respect the AMODE and RMODE attributes established by the GENMOD process. When LOADMOD executes, the RMODE specified in a relocatable MODULE file header record determines whether that file resides above or below the 16Mb line. A subsequent START command causes the module to execute with the AMODE specified in the header record.

Format

LOADMod	fn [MODULE [fm]] [(options. .[.])]] <u>Options:</u> [NOPRES] [PRES]
----------------	---

Operands

fn
is the file name of the file to be loaded into storage. The file type must be MODULE.

fm
is the file mode of the module to be loaded. If not specified, or specified as an asterisk, all your disks and directories are searched for the file.

Options

NOPRES

instructs CMS to release the storage that is occupied by the currently loaded set of non-OS programs. These programs can no longer be referenced by another program. If the module being loaded has the MAP or STR attribute, the LOADMOD command deletes from storage those programs that were previously loaded by a LOAD, INCLUDE, or LOADMOD command. NOPRES is the default.

PRES

instructs CMS to retain the storage that is occupied by the currently loaded set of programs. These programs can still be accessed via hard coded addresses, but are not executable with a START command. The loader tables are rewritten.

Usage Notes

1. You can use the LOADMOD command when you want to debug a CMS MODULE file. Since CMS MODULE files may be relocatable, you should issue the PROGMAP command after the file is loaded to help you find the location of the module in virtual storage. After issuing the PROGMAP command, you may set address stops or breakpoints before you begin execution with the START command. For example, enter the following:

```
loadmod prog1
progmap
cp per instruct range xxxxxx
start
```

2. If a MODULE file was created using the DOS option of the GENMOD command, the CMS/DOS environment must be active when it is loaded. If it was created using the OS option (the default), the CMS/DOS environment must not be active when it is loaded.
3. MODULE files created with the ALL option or with SYSTEM option may be loaded with ORIGIN=TRANS regardless of whether the CMS/DOS environment is active. If the LOADMOD command is called from a program, the loading is also done regardless of whether the CMS/DOS environment is active.
4. When in CMS SUBSET mode, a LOADMOD of a nonrelocatable program will result in return code 32.
5. Fixed transient area modules have no header record and no AMODE or RMODE attributes if the following are true:
 - they were created by the LOAD command with the ORIGIN TRANS option
 - they used the SYSTEM option from GENMOD.

The default is AMODE 24 and RMODE 24.

6. If you load a nonrelocatable program after executing a number of LOADMOD commands, the loader may detect an attempt to load over a relocatable module. There are three possible solutions:
 - Regenerate the nonrelocatable program so it will be relocatable. This is the simplest method.
 - Change the ORIGIN option of the nonrelocatable program to a lower address, thus residing at an address to be allocated at a later time.
 - Execute a LOAD or LOADMOD for the nonrelocatable program first. This will cause relocatable programs to be loaded at locations that do not conflict.
7. A nonrelocatable module created above the 16Mb line by the LOAD/INCLUDE process will not load:
 - on a System/370 mode virtual machine
 - on a 370-XA mode virtual machine with less than 16Mb of storage.

Messages and Return Codes

DMSERD257T Internal system error at address *addr* (offset *offset*)
 DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC = 31]
 DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
 DMSMOD001E No filename specified [RC = 24]
 DMSMOD002E File(s) [*fn* [*ft* [*fm*]]] not found [RC = 28]
 DMSMOD018E No load map available [RC = 40]
 DMSMOD032E Invalid filetype *ft* [RC = 24]
 DMSMOD037E Filemode *mode*(*vdev*) is accessed as read/only [RC = 36]
 DMSMOD069E Filemode *mode* not accessed [RC = 36]
 DMSMOD070E Invalid parameter *parameter* [RC = 24]
 DMSMOD104S Error *nn* reading file *fn ft fm* from disk or directory
 [RC = 31|55|70|76|99|100]
 DMSMOD109S Virtual storage capacity exceeded [RC = 104]
 DMSMOD114E *fn ft fm* not loaded; CMS/DOS environment [not] active [RC = 40
 or -0005]
 DMSMOD116S Loader table overflow [RC = 104]
 DMSMOD380E Storage at origin *addr* in use, *fn* not loaded [RC = 104]
 DMSMOD639E Error in DMSRLD routine, return code was *nnn*
 DMSMOD945E AMODE/RMODE values conflict. *file* not loaded|generated
 [RC = 68]
 DMSMOD988E Module *fn* cannot execute in 370|XA architecture [RC = 64]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in Shared File System	813
Errors in using a file	814

MACLIB

Use the MACLIB command to create and modify CMS macro libraries.

Format

MAClib	<pre> { GEN } { ADD } { REP } libname fn1 [fn2...] DEL libname membername1 [membername2...] COMP libname MAP libname [membername1 [membername2...]] [(options...)] Options: [DISK PRINT TERM STACK [FIFO LIFO] FIFO LIFO XEDIT] </pre>
---------------	--

Operands

GEN

generates a CMS macro library.

ADD

adds members to an existing macro library. No checking is done for duplicate names, entry points, or CSECTS.

REP

replaces existing members in a macro library.

DEL

deletes members from a macro library. If more than one member exists with the same name, only the first entry is deleted.

COMP

compacts a macro library.

MAP

lists certain information about the members in a macro library. Available information includes member name, size, and location relative to the beginning of the library.

libname

is the file name of a macro library. If the file already exists, it must have a file type of MACLIB; if it is being created, it is given a file type of MACLIB.

fn1 [fn2...]

are the names of the macro definition files to be used. A macro definition file must reside on a CMS disk or SFS directory and its file type must be either MACRO or COPY. Each file may contain one or more macros and must contain fixed-length, 80-character records.

membername1 [membername2...]

are the names of the macros that exist in a macro library.

MAP Options

The following options specify where the output of the MAP function is sent. Only one option may be specified. If more than one option is specified, only the first one given is used.

DISK

writes the MAP output on a CMS disk or SFS directory with the file identifier of "libname MAP A1." If no option is specified, DISK is the default.

PRINT

spools the MACLIB map to your virtual printer.

TERM

displays the MAP output at the terminal.

STACK [FIFO]**STACK LIFO**

places the MAP output in the program stack. It can be stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO

places the MAP output in the program stack. It is stacked FIFO. The options STACK, STACK FIFO, and FIFO are equivalent.

LIFO

places the MAP output in the program stack. It is stacked LIFO. This option is equivalent to STACK LIFO.

XEDIT

inserts the MAP output into the file being edited. Each map line is 130 characters and contains the following twice on the line: the member name, index, size, library file name, library file type, and library file mode. The XEDIT option does not return the header record. This option is only valid when MACLIB MAP is issued from the XEDIT environment.

Usage Notes

1. When a MACRO file is added to a MACLIB, the member name is taken from the macro prototype statement. If there is more than one macro definition in the file, each macro is written into a separate MACLIB member. If the file type is COPY and the file contains more than one macro, each macro must be preceded by a control statement of the following format:

*COPY membername

The name on the control statement is the name of the macro when it is placed in the macro library. If there is only one macro in the COPY file and it is not preceded by a COPY control statement, its name (in the macro library) is the

same as the file name of the COPY file. If there are several macro definitions in a COPY file and the first one is not preceded by a COPY control statement, the entire file is treated as one macro.

2. If you create an alias for a MACLIB, using the CREATE ALIAS command, the alias must have a file type of MACLIB.
3. If any MACRO file contains invalid records between members, the MACLIB command displays an error message and terminates. Any members read before the invalid card is encountered are already in the MACLIB. The MACLIB command ignores CATAL.S, END, and /* records when it reads MACRO files created by the ESERV program.
4. If you want a macro library searched during an assembly or compilation, you must identify it using the GLOBAL command before you begin compiling.
5. The MACLIBs distributed with the CMS system are: CMSLIB, DMSSP, DOSMACRO, OSMACRO, OSMACRO1, OSVSAM, and TSOMAC.
6. The PRINT and TERM options erase the old MAP file, if one exists.
7. If any accessed disk or directory contains a MACRO file with the same file name as the COPY file on your disk or directory accessed as A, the MACLIB's REP, ADD, and GEN functions use the MACRO version.
8. The XEDIT option is valid only when MACLIB MAP is issued from the XEDIT environment. The edited file must either be fixed format with a logical record length of 130 or variable length format. The information replaces the current line and below until the END OF FILE is reached. Then, the remaining text (if any) is inserted before the END OF FILE.
9. A return code of 4 indicates that an error occurred that did not terminate processing. Check the messages to determine the error. If the library is on a minidisk and the regeneration of the library or deleting from the library results in a library with no members, the library is erased. If the empty library is on an SFS directory, the library is maintained with a header record indicating that the library has no members. The empty library is maintained to preserve any file sharing authorities specified for them. The use of these empty libraries by other CMS commands, OS simulation macros, and other applications may produce unpredictable results.

Responses

When you enter the MACLIB MAP command with the TERM option, the names of the library members, their sizes, and their locations in the library are displayed.

```
MACRO INDEX SIZE
name  loc  size
.     .   .
.     .   .
.     .   .
```

Messages and Return Codes

DMSLBM001E No filename specified [RC = 24]
DMSLBM002W File *fn ft [fm]* not found [RC = 4 or 28]
DMSLBM003E Invalid option: *option* [RC = 24]
DMSLBM013W Member *membername* not found in library *libname* [RC = 4]
DMSLBM014E Invalid function *function* [RC = 24]
DMSLBM037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC = 36]
DMSLBM046E No library name specified [RC = 24]
DMSLBM047E No function specified [RC = 24]

MACLIB

DMSLBM056E File *fn ft [fm]* contains invalid record formats [RC=32]
DMSLBM069E Filemode *mode* not accessed [RC=36]
DMSLBM070E Invalid parameter *parameter* [RC=24]
DMSLBM104S Error *nn* reading file *fn ft fm* from disk or directory
[RC=31|55|70|76|99|100]
DMSLBM105S Error *nn* writing file *fn ft fm* on disk or directory
[RC=31|55|70|76|99|100]
DMSLBM109S Virtual storage capacity exceeded [RC=104]
DMSLBM157S MACLIB limit exceeded[, last member added was *membername*]
[RC=88]
DMSLBM167S Previous MACLIB function not finished [RC=88]
DMSLBT213W Library *fn* MACLIB not created [RC=4]
DMSLBT213W Library *fn* MACLIB not created, or erased if empty [RC=4]
DMSLBT213W Library *fn* MACLIB has no members [RC=4]
DMSLBM688E XEDIT option only valid from XEDIT environment [RC=24]
DMSLBM689E File must be F-format 130 or V-format [RC=24]
DMSLBM907T I/O error on file *fn ft fm* [RC=31|55|70|76|99|256]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814
Errors in copying a file	70
Errors in erasing a file	145
Errors in renaming a file	491

MACLIST

Use the MACLIST command to display a list of all members in a specified macro library. You can issue CMS commands against the members directly from the displayed list. In the MACLIST environment, information that is normally provided by the MACLIB command (with the MAP option) is displayed under the control of the System Product Editor. You can use XEDIT subcommands to manipulate the list itself.

Format

MACLIST MList	<p style="text-align: center;"><i>libname</i> [(options [])]</p> <p>Options: [Append] [<u>Compact</u> <u>NOCompact</u>] [PROFile <i>fn</i>]</p>
--------------------------	---

Operands

libname

is the file name of the CMS maclib for which information is to be displayed. The file must have a file type of MACLIB.

Options

Append

specifies that the list of members in this library should be appended to the existing list. This option is only valid from the MACLIST environment.

Compact

specifies that the library is to be compacted upon completion of MACLIST using the MACLIB COMP command.

NOCompact

specifies that the library is not to be compacted upon completion of MACLIST. This is the default.

PROFile *fn*

specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the MACLIST command. If not specified, a macro named PROFMLST XEDIT is invoked. For more information on the PROFMLST macro, see the usage note, "Default PF Key Settings," below.

Usage Notes

1. You can use the special commands EXECUTE and DISCARD from the MACLIST screen. The EXECUTE command allows you to issue commands that use the MACLIB members displayed by MACLIST. See "EXECUTE" on page 794 for more information. The DISCARD command allows you to erase the MACLIB members displayed by MACLIST. See "DISCARD" on page 792 for more information.

2. Tailoring the MACLIST Command Options

You can use the **DEFAULTS** command to set up options and/or override command defaults for **MACLIST**. However, the options you specify in the command line when entering the **MACLIST** command override those specified in the **DEFAULTS** command. This allows you to customize the defaults of the **MACLIST** command, yet override them when you desire. Refer to the **DEFAULTS** command description for more information.

3. Format of the List

When you invoke the **MACLIST** command you are placed in the **XEDIT** environment, editing a file "userid **MACLIST** A0." A sample **MACLIST** screen is shown in the "Examples" section. Each line in this file contains:

- a command area
- member name
- index
- number of records in the member
- library file name, library file type, and library file mode.

The full power of **XEDIT** is available to you while you issue commands against the list of members. For example, you may want to use **XEDIT** subcommands to scroll through the list of members to locate a particular member, etc.

However, some **XEDIT** subcommands are inappropriate in this environment. Subcommands that alter the format or the contents of "userid **MACLIST**" (for example, **SET TRUNC**, **SET FTYPE**, or **SET LINEND**) may cause unpredictable results.

4. Entering CMS commands from MACLIST:

Begin CMS commands with "CMS" to prevent **XEDIT** from decoding the command. This prevents CMS commands from being mistaken as **XEDIT** subcommands.

5. Issuing Commands From the List

On a full screen display, you can issue commands directly from the line on which a member name is displayed. You do this by moving the cursor to the line that describes the member to be used by the command, typing the command in the space provided to the left of the member name, and then pressing the **ENTER** key to execute the command.

If a command is longer than the command space provided on the screen, just continue typing over the information in the line. You may type over the entire line displayed, up to column 79.

The **ENTER** key is set to **EXECUTE**, which is described in "EXECUTE" on page 794. When you press the **ENTER** key, all commands typed on one screen are executed, and the screen is restored to its previous state. However, the list is updated to reflect the current status of the members (see "Responses").

You may want to enter commands from the **MACLIST** command line before executing commands that are typed on the list. To do this, move the cursor to the command line by using the **PF12** key (instead of the **ENTER** key). After typing a command on the command line and pressing **ENTER**, you can use **PF12** to move the cursor back to its previous position on the list.

Another way to issue commands that make use of the members displayed is to issue EXECUTE from the MACLIST command line. A complete description of EXECUTE is in "EXECUTE" on page 794.

6. Using Symbols as Part of a Command

Symbols can be used to represent operands in the command to be executed. They can be used in the commands typed on the screen, or as part of the command in EXECUTE (on the command line). Symbols are needed if the command to be executed has operands or options that follow the file ID. Examples of using symbols are in the "Examples" section, below.

The following symbols can be used:

- / means the *libname libtype libmode* (MEMBER *membername* that is displayed on the line.
- /l means the library filename displayed on the line.
- /t means the library filetype displayed on the line.
- /m means the library filemode displayed on the line.
- /n means the member name displayed on the line.
- /o means execute the line as is, and do not append anything.

Any combinations of symbols can be used. For example:

- /l /t means the library filename followed by library filetype.
- /lt means the library filename followed by library filetype.
- /tl means the library filetype followed by library filename.
- /ltm (MEMBER /n is equivalent to / alone.

After the command(s) have been executed, EXECUTE updates the status of the list with any changed information and uses asterisks and return codes to indicate command completion. See *Responses* above.

7. Special Symbols Used Alone

The following special symbols can be typed alone on the lines of the MACLIST display. They have the following meanings:

- = means execute the previous command for this member. Commands are executed starting at the top of the screen. For example, suppose you enter the DISCARD command on the top line. You can then type an equal sign on any other line(s). Those member preceded by equal signs are discarded when the EXECUTE command is entered (from the command line or by pressing the ENTER key).
- ? means display the last command executed. The command is displayed on the line in which the ? is entered.
- / means make this line the current line. (On the MACLIST screen, the current line is the first member name on the screen.)

8. If a member is a duplicate name and it is not the first one found in the maclib, you cannot issue any CMS commands, XEDIT subcommands, or any special symbols (=, ?, and /) for that member.

9. Default Key Settings

MACLIST

The PROFMLST XEDIT macro is executed when the MACLIST command is invoked, unless you specified a different macro as an option in the MACLIST command. It sets the keys to the following values:

ENTER	Execute command(s) typed on member line(s) or on the command line. (The ENTER key is set by the XEDIT subcommand, SET ENTER IGNORE MACRO EXECUTE).
PF 1 Help	Display MACLIST command description.
PF 2 Refresh	Update the list to indicate new members, deleted members, etc., using the same parameters as those specified when MACLIST was invoked.
PF 3 Quit	Exit from MACLIST.
PF 4 Sort(name)	Sorts by member name.
PF 5 Sort(index)	Sorts by index, largest first.
PF 6 Sort(size)	Sorts by size, largest first.
PF 7 Backward	Scroll back one screen.
PF 8 Forward	Scroll forward one screen.
PF 9 Fl /n	Issue the command FILELIST /n * * at the cursor, so that a list is displayed, containing all files that have a file name that is the same as the member name displayed on the line containing the cursor. (all file types and file modes).
PF 10	Not assigned.
PF 11 XEDIT	Edit the member where the cursor is placed.
PF 12 Cursor	If the cursor is in the file area, move it to the command line. If the cursor is on the command line, move it back to its previous location in the file (or to the current line).

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFMLST XEDIT macro sets synonyms that you can use to sort your MACLIST list. The synonyms are:

SINDEX	Sorts the list by index (greatest to least) within a library.
SLIB	Sorts the list alphabetically by library file ID, then by member name and index.
SNAME	Sorts the list alphabetically by member name and then by library file ID and index.
SSIZE	Sorts the list by member size (number of records, greatest to least).

10. If you want to issue MACLIST from an exec program, you should precede it with the EXEC command; that is, specify

```
exec maclist
```

Examples

The following MACLIST screen was created by issuing the MACLIST command as follows.

```
maclist mylib
```

Note that the members are sorted alphabetically by member name. Members with the same name are then sorted by index number (least to greatest).

```

FARRELL MACLIST A0 V 130 Trunc=130 Size=18 Line=1 Col=1 Alt=0
Cmd Member name Index Records Library name Library type Mode
CAUTION 190 6 MYLIB MACLIB A1
FAST 240 25 MYLIB MACLIB A1
FORWARD 613 57 MYLIB MACLIB A1
GO 197 25 MYLIB MACLIB A1
GO 615 25 MYLIB MACLIB A1
LTURN 546 55 MYLIB MACLIB A1
NEUTRAL 266 5 MYLIB MACLIB A1
PARK 602 4 MYLIB MACLIB A1
REVERSE 272 118 MYLIB MACLIB A1
RTURN 524 21 MYLIB MACLIB A1
SKID 391 43 MYLIB MACLIB A1
SLOW 671 61 MYLIB MACLIB A1
SLOWER 435 5 MYLIB MACLIB A1
SLOWEST 441 82 MYLIB MACLIB A1
SPEED 2 132 MYLIB MACLIB A1
STOP 607 5 MYLIB MACLIB A1
SWERVE 223 16 MYLIB MACLIB A1
1= Help 2= Refresh 3= Quit 4= Sort(name) 5= Sort(index) 6= Sort(size)
7= Backward 8= Forward 9= FL /n 10= 11= XEDIT 12= Cursor
====>
X E D I T 1 File
    
```

Figure 10. Sample MACLIST Screen

The following examples show how symbols can be used to represent operands in a command. The values substituted for the symbols and the resulting command are shown. In each case, the command can be entered in either of the following ways:

- Typed in the “Cmd” area of the screen. The command is executed either by entering EXECUTE on the XEDIT command line and then pressing ENTER, or simply by pressing ENTER.
- Entered from the XEDIT command line, as an operand of EXECUTE (in the form “EXECUTE lines command”).

If a symbol is not specified, the *libname*, *libtype*, *libmode*, and (MEMBER *membername*) are appended automatically to the command.

COMMAND	RESULTING COMMAND
discard /ltm (member yield	MACLIB DEL MYLIB (YIELD
listfile /lt *	LISTFILE MYLIB MACLIB *
copyfile /ltm /l oldlib /m	COPYFILE MYLIB MACLIB A1 MYLIB OLDLIB A1

MACLIST

Responses

When a command is executed, one of the following symbols is displayed in the "Cmd" space to the left of the file for which the command was executed.

- * Means the command executed successfully (RC=0).
- *n Is the return code from the command executed (RC=n).
- *? Means that the command was an unknown CP/CMS command (RC=-3).

The following responses can also appear directly on the MACLIST screen:

- * Library 'libname libtype libmode' not found. *
- * membername has been discarded.
- * membername ** Member is a duplicate entry. Invalid for EXECUTE **
- * membername ** Member is no longer in the library. **
- Member *membername* has been discarded.

Messages and Return Codes

- DMSWML002E File *fn* MACLIB * not found [RC=28]
- DMSWML651E APPEND must be issued from MACLIST [RC=40]
- DMSWML668E The APPEND option must be specified alone [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

MAKEBUF

Use the MAKEBUF command to create a new buffer within the program stack.

Format

MAKEBUF	
---------	--

Usage Notes

1. When you issue a MAKEBUF command, CMS returns as a return code the number of the program stack buffer just created. If you issue a MAKEBUF command in an exec that has the &ERROR statement in effect, the MAKEBUF return code causes the &ERROR statement to execute.
2. Use the WAITRD function to read lines from the buffers the MAKEBUF command creates. WAITRD first reads lines from the most recently created buffer. When the most recent buffer is exhausted, WAITRD reads the next most recent buffer. When all program stack buffers are exhausted, WAITRD reads from the terminal input buffer.

MODMAP

Use the MODMAP command to display the load map associated with the specified MODULE file.

Format

MODmap	<i>fn</i>
--------	-----------

Operands

fn

is the file name of the MODULE file whose load map is to be displayed. The file type of the file must be MODULE; all of your accessed disks and directories are searched for the specified file.

Usage Notes

1. You cannot issue a MODMAP command for modules that are CMS transient area modules or that have been created with the NOMAP option of the GENMOD command.
2. When in CMS SUBSET mode, a MODMAP that requires a LOADMOD into the user area will receive a return code 32 from LOADMOD.

Responses

The load map associated with the file is displayed at the terminal, in the format:

```
name      location
.         .
.         .
.         .
```

Messages and Return Codes

DMSMDP001E No filename specified [RC = 24]
 DMSMDP070E Invalid parameter *parameter* [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

MOREHELP

Use the MOREHELP command to obtain either additional or related information about the last valid HELP command you issued. This command is particularly helpful if you use a linemode terminal or if you cannot use a PF key to obtain DETAIL or RELATED information.

Format

MOREhelp	<pre>[([optionA] [optionB] [])]</pre> <p>OptionA: [<u>DETail</u> BRIEf RELated]</p> <p>OptionB: [<u>ALL</u>] [DESCript] [FORMat] [PARMs] [OPTions] [NOTEs] [ERRors]</p>
-----------------	---

OptionA

DETail

displays the DETAIL layer of the HELP file. The amount of DETAIL information displayed is determined by your DEFAULTS selection of the subsetting options. The subsetting options are ALL or DESCRIPT, FORMAT, PARMs, OPTions, NOTEs, and ERRors. The options DESCRIPT, FORMAT, PARMs, OPTions, NOTEs, and ERRors can be specified in any combination. For Options A, the DETAIL option is the default.

BRIEf

displays the BRIEF layer of the HELP file. If the BRIEF layer is not available, then you will see the next available layer of information.

RELated

displays the RELATED layer of the HELP file, if available.

OptionB

The following options can be specified with the DETAIL operand.

ALL

displays all the DETAIL information of the HELP file. This includes the DESCRIPT, FORMAT, PARMs, OPTions, NOTEs, and ERRors sections. It does not include the BRIEf and the RELated sections. For Options B, the ALL option is the default.

DESCript

displays the description information of the HELP file.

FORMat

displays the format information (the syntax).

MOREHELP

PARMs

displays the parameter section (explanation of the operands).

OPTions

displays the options section (a list of available options with a brief description).

NOTEs

displays the usage notes and example sections.

ERRors

displays the error messages and response sections.

Usage Notes

1. The **DETAIL** option displays the specified subset information (as preset by the **DEFAULTS** command). The initial default is set to display all six subset sections. If the file does not contain the **DETAIL** subset you have specified, then a message and the next available layer of **HELP** information will be displayed.
2. Additional information is based upon the last valid **HELP** command you entered. If you have not entered a valid **HELP** command, message **DMSMOR353E**, "No previous **HELP** command has been entered," will be displayed.
3. **RELATED HELP** information is available for a limited number of **HELP** files. However, if you wish to tailor your own **HELP** files to include this option, you may do so. Refer to "Tailoring the **HELP** facility" in the *VM/SP CMS User's Guide*.
4. If there is no **RELATED HELP** information available, you will see a warning message. No information will be displayed.
5. If you want to issue **MOREHELP** from an exec program, you should precede it with the **EXEC** command; that is, specify
`exec morehelp`

Messages and Return Codes

- DMSMOR353E** No previous **HELP** command has been entered. Please enter **HELP MOREHELP** for information on the **MOREHELP** command. [RC=4]
- DMSMOR354E** **RELATED** information is not available for the last **HELP** command entered. [RC=32]
- DMSMOR639E** Error in *routine* routine; return code was *xx*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

MOVEFILE

Use the MOVEFILE command to move data from any device supported by VM/SP to any other device supported by VM/SP.

Format

MOVEfile	$\left[\begin{array}{cc} \textit{inddname} & \textit{outddname} \\ \underline{\text{INMOVE}} & \underline{\text{OUTMOVE}} \end{array} \right] [(\text{PDS } [])]$
-----------------	---

Operands

inddname

is the ddname representing the input file definition. If ddname is not specified, the default input ddname, INMOVE, is used.

outddname

is the ddname representing the output file definition. If ddname is not specified, the default output ddname, OUTMOVE, is used.

Option

PDS

moves each of the members of the CMS MACLIB or TXTLIB or of an OS partitioned data set into a separate CMS file. Each CMS file has a file name equal to the member name and a file type equal to the file type of the output file definition.

Usage Notes

1. Use the FILEDEF command to provide file definitions for the ddnames used in the MOVEFILE command. If you use the ddnames INMOVE and OUTMOVE on the FILEDEF commands, then you need not specify them on the MOVEFILE command line. For example:

```
filedef inmove disk sys1 maclib b (member stow
filedef outmove disk stow macro
movefile
```

copies the member STOW from the OS partitioned data set SYS1.MACLIB into the CMS file STOW MACRO.

If you enter:

```
filedef indd reader
filedef outdd printer
movefile indd outdd
```

a file is moved from your virtual reader to your virtual printer.

2. To copy an entire OS partitioned data set into individual CMS files, you could enter:

```
filedef test2 disk sys1 maclib b
filedef macro disk
movefile test2 macro (pds
```

These commands copy members from the OS partitioned data set SYS1.MACLIB or the CMS file SYS1 MACLIB into separate files, each with a file name equal to the member name and a file type of MACRO. Note that the output ddname was not specified in full, so that CMS assigned the default file definition (FILE ddname).

3. You cannot copy VSAM data sets with the MOVEFILE command.
4. The MOVEFILE command does not support data containing spanned records. Use of spanned records results in the error message DMSSOP036E and an error code of 7.
5. To copy an entire partitioned data set into another partitioned data set, use the COPYFILE command. If an attempt is made to use the MOVEFILE command without the PDS option for a partitioned data set, only the first member is copied and an end-of-file condition results. The resultant output file will contain all input records, including the header, until the end of the first member.
6. When using the MOVEFILE command to move members from CMS maclibs, note that each member is followed by a // record, which is a maclib delimiter. You can edit the file to delete the // record.
7. If you use the MOVEFILE command and FILEDEF command with the options DISP MOD and RECFM FB to add a file to the end of an existing OS simulated file, the user should erase the end-of-file mark at the end of the existing file. The end-of-file mark will be present only if the last physical record written was a short block. In addition, during multi-volume tape processing, DISP MOD is only valid with the tape currently mounted; no volume switching will be done.
8. The following record formats are supported for DOS files on FBA devices: fixed, fixed blocked, variable, variable blocked, and undefined. The FILEDEF for the input file must specify at least the RECFM and BLOCK; for fixed block files the LRECL must also be specified. Do not issue "SET DOS ON". If you do, MOVEFILE will result in an error message.
9. When copying a variable length data set (RECFM=V or VB) from an OS disk to a CMS disk or SFS directory, the logical record length (LRECL) of the file that is created is equal to the size of the largest record in the data set being copied. If the file that is being created has a file mode of 4, the logical record length will be equal to the LRECL of the largest record plus 8 bytes. The actual LRECL of the new file can be determined by using the CMS LISTFILE command.
10. For OS compatibility of the output tape labels, you must specify LRECL and BLOCK/BLKSIZE in your output FILEDEF.
11. Any attempt to move an empty OS data set that has not been closed will cause unpredictable results.

Default Device Attributes:

If a record format (RECFM), blocksize (BLOCK), and logical record length (LRECL) are specified on the FILEDEF command, these values are used in the data control block (DCB) defining the characteristics of the move operation. If the FILEDEF was issued without a record format or blocksize specified, these values are determined according to the defaults listed in Figure 11. If the blocksize was not specified, the default blocksize is used. If the logical record length was not specified, the default logical record length is determined as follows: for an F or U record

format, the logical record length equals the blocksize; for a V record format, the logical record length equals the blocksize minus 4.

Device	Input ddname		Output ddname	
	RECFM	Blocksize	RECFM	Blocksize
Card Reader	F	80	NA ²	NA ²
Card Punch	NA ²	NA ²	F	80
Printer	NA ²	NA ²	U	132
Terminal	U	130	U	130
Tape ¹	U	3600	RECFM of input ddname	Blocksize of input ddname
Disk or directory	RECFM of file	Blocksize of file	RECFM of input ddname	Blocksize of input ddname
Dummy	NA ¹	NA ²	RECFM of input ddname	Blocksize of input ddname
¹ If the default record format and blocksize are used in a tape-to-tape move operation and an input record is greater than 3600 bytes, it is truncated to 3600 bytes on the output tape. ² not applicable.				

Figure 11. Default Device Attributes for MOVEFILE Command

Responses

DMSMVE225I PDS member *membername* moved

The specified member of an OS partitioned data set was moved successfully to a CMS file. This response is issued for each member moved when you use the PDS option.

DMSMVE226I End of PDS move

The last member of the partitioned data set was moved successfully to a CMS file.

DMSMVE706I Terminal input; type null line for end of data

The input ddname in the MOVEFILE specified a device type of terminal. This message requests the input data; a null line terminates input.

DMSMVE708I File *fn ddname A1* assumed for DDNAME *ddname*

No file definition is in effect for a ddname specified on the MOVEFILE command. The MOVEFILE issues the default FILEDEF command:

FILEDEF ddname DISK FILE ddname A1

If file ddname does not exist for the input file, MOVEFILE terminates processing.

Messages and Return Codes

DMSMVE002E File[(s)] [*fn* [*ft* [*fm*]]] not found [RC = 28]
DMSMVE003E Invalid option: *option* [RC = 24]
DMSMVE037E Output filemode *mode*[(*vdev*)] is accessed as read/only [RC = 36]
DMSMVE041E Input and output files are the same [RC = 40]
DMSMVE069E Output filemode *mode* is not accessed [RC = 36]
DMSMVE070E Invalid parameter *parameter* [RC = 24]
DMSMVE075E Device *devtype* invalid for {input|output} [RC = 40]
DMSMVE086E Invalid DDNAME *ddname* [RC = 24]
DMSMVE127S Unsupported device for DDNAME [RC = 100]
DMSMVE128S I/O error on input after reading *nnn* records; input error code on DDNAME [RC = 100]
DMSMVE129S I/O error on output writing record number *nnnn*; output error code on DDNAME [RC = 100]
DMSMVE130S Blocksize on V format file *ddname* is less than 9 [RC = 88]
DMSMVE232E Invalid RECFM--spanned records not supported [RC = 88]
DMSMVG089E Open error code *nn* on {*fn*|SYSaaa|*tapn*} [RC = 36]

NAMEFIND

Use the NAMEFIND command to display information from a names file, or to place that information in the program stack (for use by an exec or other program).

A names file has a file type of NAMES and must be in the format described in the usage note below, "Format of a Names File." A "userid NAMES" file is a special names file, used by the NAMES, NOTE, SENDFILE, RECEIVE, and TELL commands, that makes it easier for you to communicate with other computer users. You can use the NAMES command to create a "userid NAMES" file. NAMEFIND searches a "userid NAMES" file, unless a different file name is specified.

Format

NAMEFInd	<pre><i>:tag value</i> [<i>:tag</i> [<i>value</i>]]... [(options... [])]</pre> <p>Options:</p> <pre>[STACK [<i>n</i> * <u>1</u>] [FIFO LIFO]]</pre> <pre>[FIFO [<i>n</i> * <u>1</u>]]</pre> <pre>[LIFO [<i>n</i> * <u>1</u>]]</pre> <pre>[TYPE [<i>n</i> * <u>1</u>]]</pre> <pre>[FILE <i>fn</i>] [LINenum] [START <i>recnum</i>]</pre> <pre>[SIze [<i>n</i> * <u>8</u>]] [XEDIT]</pre>
-----------------	---

Operands

:tag

is a tag in a names file. You can specify multiple tags in a NAMEFIND command. The maximum length of a tag is 255. For more information on tags, see the usage note, "Format of a Names File."

value

is the value of a tag in a names file. The maximum length of a value is 255.

Options

STACK [*n*] [**FIFO**]

STACK [*n*] **LIFO**

means that information from the number of entries specified (*n*) that meet the search criteria is placed in the program stack, rather than being displayed at the terminal. The number (*n*) specified is the number of entries containing matching information. Valid values for *n* are 1 to 99999999 or *. If *n* is omitted, the default is one (1). If an asterisk (*) is specified, information from all the entries meeting the search criteria is stacked. If more than 8 digits are specified for *n*,

the value is truncated on the right. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO *n*

specifies that the information is placed in the console stack. Valid values for *n* are 1 to 99999999 or *. If more than 8 digits are specified for *n*, the value is truncated on the right. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO *n*

specifies that the information is placed in the console stack rather than being displayed at the terminal. The information is stacked LIFO (last in first out). Valid values for *n* are 1 to 99999999 or *. If more than 8 digits are specified for *n*, the value is truncated on the right. This option is equivalent to STACK LIFO.

TYPE *n*

means that information from the number of entries specified (*n*) that meet the search criteria is displayed at the terminal. The number (*n*) specified is the number of entries containing matching information. Valid values for *n* are 1 to 99999999 or *. If *n* is omitted, the default is one (1). If an asterisk (*) is specified, information from all the entries meeting the search criteria is displayed. If more than 8 digits are specified for *n*, the value is truncated on the right. This option is the default.

FILE *fn*

specifies a file whose file name is "fn" and whose file type is "NAMES." This option allows you to use NAMEFIND to search a names file whose file name is something other than your user ID. If this option is not specified, the file "userid NAMES *" is searched.

LINenum

requests that the record number of the beginning of the entry be displayed or stacked. It is displayed or stacked before any of the other information. The record number returned has 9 digits and a value of 1 to 99999999.

START *recnum*

specifies that the search is to begin at the *recnum* record of the file. Valid values for *recnum* are 1 to 99999999 or *. If more than 8 digits are specified for *n*, the value is truncated on the right.

SIZE *n*

specifies the maximum size of a buffer where a names file is kept. The size of the buffer is *n*, where *n* is in 1024-character units. Valid values for *n* are 0 to 99999999 or *. If zero (0) is specified, no buffer is used, and the names file is read into storage each time NAMEFIND is invoked. If an asterisk (*) is specified, the buffer is as large as the names file requires. If more than 8 digits are specified for *n*, the value is truncated on the right. If no SIZE option is specified, SIZE 8 (8192 characters) is the default. This represents the maximum size of the buffer. (If the names file is smaller than 8192 characters, a smaller buffer is used.) This option improves the performance of NAMEFIND when a names file is large. For more information on its use, see the usage note, "Using the SIZE Option," below.

XEDIT

specifies that the information should be read from the file in storage rather than from a minidisk or directory. This option is only valid when NAMEFIND is issued from the XEDIT environment.

Usage Notes

1. Format of a Names File

A names file is a collection of entries, with each entry identified by a "nickname." A nickname tag plus a series of other tags with associated values make up an entry.

A special names file is one whose file ID is "userid NAMES," which can be created using the NAMES command. A "userid NAMES" file contains entries for other computer users and entries for lists of users. An entry contains the information necessary to communicate with that person. Once you create a "userid NAMES" file, you can prepare notes for and send files and messages to other people just by using their "nicknames" as operands in the NOTE, SENDFILE, and TELL commands. The tags in each entry supply the additional information required to perform these functions.

You can add, remove, or change entries in the "userid NAMES" file either by using the NAMES command (which displays a menu), or by editing the "userid NAMES" file directly. (The NAMES command can be used only for a file whose file ID is "userid NAMES.")

A sample "userid NAMES" file is shown below, in the "Examples" section.

Format of Entries in a Names File:

The format of data lines in a names file is as follows:

```
:tag.value [:tag.value...]
```

The value need not be on the same record as its tag and can continue onto the next record.

The only tag that is required is a :NICK tag:

```
:NICK.nickname
```

This is the primary tag, one for each entry. It identifies the beginning of an entry and must be the first word on a line.

Any tags that follow relate to the preceding :NICK tag. The maximum number of tags with values for a given :NICK entry is 64. Therefore, between :NICK entries, you can have from zero to 63 tags.

In addition,

```
*      An asterisk in column one begins a comment line.
.*     A period in column one and asterisk in column two
       begin a comment line.
Blank lines are ignored.
```

2. How NAMEFIND Searches a Names File

NAMEFIND

When you issue a NAMEFIND command, each tag specified with a value is a search tag. NAMEFIND searches until all search tags are found in an entry. Each tag specified without a value is a "return" tag, whose value is returned. If no return tags are specified, the entire entry is displayed or stacked.

Given the "userid NAMES" file shown in the "Examples" section below, the command

```
NAMEFIND :NICK SNOW :NAME :PHONE
```

would display:

```
Snow White  
ZZZ-ZZZZ
```

(:NICK SNOW is the search tag. :NAME and :PHONE are the return tags.)

If you specify two or more identical 'return' tags, a value for each of the tags is returned. A null value is returned to a 'return' tag if an identical 'return' tag does not exist in the Names file. Given the "userid NAMES" file used in the above example, the command:

```
NAMEFIND :NICK SNOW :NAME :NAME :NAME
```

displays:

```
Snow White
```

Ready;

(:NICK SNOW is the search tag and :NAME, :NAME, and :NAME are identical return tags).

You can specify the tag ":LIST" to display all the names in a list. For example, the command

```
NAMEFIND :NICK DWARFS :LIST
```

would display:

```
SNOOZY DUMMY BOSS SMILEY GROUCHY SNIFFLES WISTFUL
```

You could then issue NAMEFIND for each of the names in the list shown above, specifying the return tag :USERID to retrieve the user ID of each person.

You need know the value of only one unique tag in an entry for that entry to be located. The tags specified without values determine the information that is displayed (or stacked). For example, the command

```
NAMEFIND :USERID QUEEN :NICK
```

would display:

```
WITCH
```

If duplicate entries exist in a names file, only the first is found, unless an option value (n) greater than one is specified. If duplicate :NICK entries are submitted from the NAMES menu (which is displayed with the NAMES command), a warning message is displayed.

Case and multiple blanks are ignored during the search. Case and multiple blanks in tag values are preserved when the values are displayed or stacked.

3. Tags in a "userid NAMES" File

The CMS commands that reference a "userid NAMES" file are NOTE, SENDFILE, TELL, and RECEIVE. These commands make use of the tags

described below. Fields that correspond to these tags appear on the NAMES menu. You can also add other tags to the file (for example, for use by other applications).

:NICK.nickname

This is the primary tag, one for each person or list in the file. It identifies the beginning of an entry and must be the first word on a line.

You should have a :NICK entry for yourself, because the tags that supply your address, phone number, etc., are used by the NOTE command to generate note headings.

All of the following tags relate to the preceding :NICK tag. (Not all tags are required for each entry; however, the CMS commands that reference the "userid NAMES" file make use of the following tags.)

:USERID.userid

specifies the user ID of the preceding :NICK entry. This tag is required for communicating with this user via NOTE, SENDFILE, and TELL. If no :USERID tag is specified, the nickname is just an entry (for an address list, or perhaps a name of someone who does not use a computer).

:NODE.node

specifies the node of the preceding :NICK entry. If no node is specified, the default node is your node.

:NOTEBOOK.filename

is the name of a file whose file type is NOTEBOOK, in which notes (prepared by the NOTE command) sent to or received from this person are kept. See the NOTE command for more information on keeping notes.

:NAME.name

is the person's real name.

:PHONE.phone number

is the person's phone number.

```
:ADDR.address
```

is the person's postal address. Semicolons (;) in the tag's value separate the lines of the address. They do not appear in the header of a note (prepared by the NOTE command).

```
:LIST.[name...]
```

is a list of names. If a name in the list is not a nickname in the "userid NAMES" file, it is assumed to be a user ID on the sender's computer. A name can also be specified as "userid AT node," just as it can in the NOTE, SENDFILE, and TELL commands. The nickname specified on the associated :NICK tag can be the nickname for the whole list, or it can be the nickname for one user.

4. Private Resource Processing

Private resource registrations and user ID authorizations are contained in a special names file called "\$SERVER\$ NAMES." The information collected in the \$SERVER\$ NAMES file is used to validate private resource conversation requests. See the *VM/SP Connectivity Planning, Administration, and Operation* book for additional information on using the "\$SERVER\$ NAMES" file and private resource processing.

5. CMS Communications Directory Processing

Symbolic destination name definitions, and access security information are contained in special names files called communications directories. The default system-level communications directory is called SCOMDIR NAMES; the user-level communications directory is called UCOMDIR NAMES. The information collected in the communications directories is used to resolve symbolic destination names when CMS communications directory processing is enabled. Communications directory processing is enabled by the SET COMDIR command. See the *VM/SP Connectivity Planning, Administration, and Operation* book for additional information on using CMS communications directories.

6. Using the SIZE Option

When NAMEFIND is invoked, the names file is read into a buffer in virtual storage. It is kept in this buffer instead of being read from a minidisk or directory each time NAMEFIND is invoked. In following invocations of NAMEFIND, characteristics of the NAMES file such as reacl, file mode, and creation date and time, are compared with those of the file in the buffer to determine if the file has been changed. If so, the changed file is read into the buffer. The NAMES file is always read into storage when the SIZE option is specified, regardless of whether the characteristics of the NAMES file have changed or not. The CMS commands - NOTE, RECEIVE, SENDFILE, and TELL - all invoke the NAMEFIND command to search a names file. Having a names file kept in a buffer improves performance of these commands, particularly if the file is large.

When the XEDIT option is specified, the SIZE option is ignored because the file is already in storage.

The maximum buffer size is the value of the last SIZE option specified. If you do not specify the SIZE option, the default buffer size is SIZE 8 (8192 characters). If a names file is too large to fit in the buffer, you can increase the size of the buffer accordingly. Naturally, it can also be decreased to conserve virtual storage. However, if the names file is larger than the size (n) allocated for the buffer, NAMEFIND reads as much of the file as will fit into the buffer, and then reads the rest from the minidisk or directory. By specifying "NAMEFIND (SIZE *)" (a good candidate for your PROFILE EXEC), the buffer uses as much storage as is needed to contain a names file, no more and no less.

7. The total length of all tags and values specified may not exceed 255.
8. NAMEFIND uses the extended plist for processing the *:tag.value* parameter. If you are calling NAMEFIND from an assembler language program and using *:tag.value*, you should supply an extended plist. The *VM/SP Application Development Guide for CMS* book has more information on how an assembler language program can supply an extended plist.

Note: When you are working with a names file, and the NODE tag in the NAMES file indicates that the user is on another processor, then you must also specify a LOCALID tag.

Examples

The following is a sample "userid NAMES" file:

```

:nick.SNOW      :userid.SNOWHITE :node.FOREST
                :name.Snow White           :phone.ZZZ-ZZZZ
                :addr.Forest Primeval
:nick.SNOOZY    :userid.SNOOZY   :node.COTTAGE
                :name.I. M. Dozing       :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.DUMMY     :userid.DUMMY    :node.COTTAGE
                :name.S. A. What         :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.BOSS      :userid.BOSS     :node.COTTAGE
                :name.T.O.P. Banana     :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.SNIFFLES  :userid.SNIFFLES :node.COTTAGE
                :name.A. H. Choo        :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.GROUCHY   :userid.GROUCHY :node.COTTAGE
                :name.E. B. Scrooge     :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.SMILEY    :userid.SMILEY  :node.COTTAGE
                :name.H. A. Haas        :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.WISTFUL   :userid.WISTFUL  :node.COTTAGE
                :name.R. U. Shy         :phone.777-7777
                :addr.Dwarf Cottage;Forest
:nick.WITCH     :userid.QUEEN   :node.CASTLE
                :name.Bad Queen         :phone.UGLY-1111
                :addr.Vanity Lane;Mirror City
:nick.GORGEOUS  :userid.PRINCE  :node.ATLARGE :notebook.PRIVATE
                :name.Prince Charming   :phone.Area 111 111-1111
                :LOCALID.frog
:nick.DWARFS    :list. SNOOZY DUMMY BOSS SMILEY GROUCHY SNIFFLES WISTFUL

```

Figure 12. Sample 'userid NAMES' File

Messages and Return Codes

```

DMSNAM002E Files [fn [ft [fm]]] not found [RC=28]
DMSNAM003E Invalid option: option [RC=24]
DMSNAM029E Invalid parameter parameter in the option option field [RC=24]
DMSNAM104S Error nn reading file fn ft fm from {disk or directory|XEDIT}
           [RC=100]
DMSNAM156E Record nnn not found--the file fn ft fm has only nnn records
           [RC=32]
DMSNAM618E NUCEXT failed [RC=nm]
DMSNAM621E Bad plist: message [RC=24]
DMSNAM622E Insufficient free storage for NAMEFIND [RC=rc]
DMSNAM622W Insufficient free storage for NAMEFIND buffer, processing
           continues
DMSNAM633E Too many tags were encountered--maximum is 64 per line
           [RC=88]
DMSNAM633W Returned values were truncated [RC=88]
DMSNAM634E No value to search for was specified [RC=24]
DMSNAM635I No entries were found that matched your search criteria
DMSNAM637E Missing value for the option option [RC=24]
DMSNAM688E XEDIT option only valid from XEDIT environment [RC=24]

```

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

NAMES

Use the NAMES command to display a menu from which you can create, change, and remove entries in a *userid* NAMES file. The menu can be used only on a display terminal.

Format

NAMES	<i>[nickname]</i>
-------	-------------------

Operands

nickname

is the name assigned to an entry in a *userid* NAMES file. If you specify a nickname, the NAMES menu is displayed with all the information from that entry (if the entry exists) filled in on the menu. You can then examine or change the values in that entry. For example, you might want to update someone's address or phone number.

If the entry does not exist, the menu is displayed with only the nickname field filled in (with the nickname you specified) truncated to 8 characters. You can then fill in the other fields to add a new entry to the NAMES file.

If you invoke NAMES without specifying a nickname, the menu is displayed with all fields left blank. You can then fill in the blanks on the menu to create a new entry, or you can scroll through the names file.

Usage Notes

1. What Is a *userid* NAMES File?

A *userid* NAMES file (where *userid* is your user ID) is a collection of information about other computer users with whom you communicate. An entry in this file is all the information associated with a particular nickname.

Having a "*userid* NAMES" file makes it easier for you to communicate with other users, because you can assign nicknames to them. You can then prepare notes for and send files and messages to other users by using their nicknames as operands in the NOTE, SENDFILE, and TELL commands.

You can also create an entry for a list of names. In this case, the nickname refers to all the users in the list. This makes it possible to send notes, files, or messages to everyone on the list by issuing the appropriate command only once.

2. Entering Information on the NAMES Menu for the "*userid* NAMES" file

The NAMES menu helps you to create and edit a *userid* NAMES file. All of the information you type on one menu is an entry in the file. You fill in the fields on the menu and press a PF key to create, display, and/or change your names file. The PF key functions are described in the usage note, "PF Key Settings on the NAMES Menu."

The following list describes the various fields on the menu and explains the information you type in. Refer to the sample menus in the "Examples" section, below, to see the location of the fields on the menu.

Nickname:

is any name you choose to represent a single user or a list of users. An example of each is shown in the "Examples" section, below. Once an entry is created, the nickname is the only piece of

information you need to communicate with this user (using the NOTE, SENDFILE, or TELL commands).

You should create an entry for *yourself*, because the fields that contain your mailing address, phone number, etc., are used by the NOTE command to generate headings.

It is recommended that the nickname not contain any character that is defined as one of the special logical characters, such as CHARDEL (character delete symbol), LINEDEL (line delete symbol), LINEND (line end symbol), ESCAPE (escape character), or TABCHAR (tab character). For further information, consult the *VM/SP CP General User Command Reference*.

Userid:

is the user ID of the person whose nickname you specified. You can leave this field blank if the nickname represents a list, that is, if the List of Names field is filled in. However, if the nickname represents a list and you also specify a user ID, the note is also sent to this user ID.

You can also leave this field blank if you want the entry to contain information about a person, but you do not intend to communicate with him via the computer. You might choose to do this if you're using the NAMES file simply to compile an address list.

Node:

is the node of the person whose nickname you specified. If not specified, the default node is the one on which this names file exists. You can leave this field blank if the nickname represents a list.

Notebook:

is the file name of a file whose file type is NOTEBOOK, in which notes (prepared by the NOTE command) sent to or received from this person are to be kept. You can leave this field blank if you want all incoming and outgoing notes saved in the default notebook file, ALL NOTEBOOK.

Name:

is the name of the person whose nickname you specified. You can leave this field blank if the nickname represents a list.

Phone:

is the phone number(s) of the person whose nickname you specified. You can leave this field blank if the nickname represents a list.

Address:

is the address of the person whose nickname you specified. You can leave this field blank if the nickname represents a list.

List of names:

is the names of the people in a list, when the nickname represents the name of this list. The names of the people in the list can be specified in the following ways: as a nickname of an entry in the names file; as a user ID of a user who shares your computer; or in the form *userid AT node*. Each time you send a note, a file, or a message to the nickname specified, it will go to everyone on this list. A sample entry for a list of names is shown in the "Examples" section, below.

Tag:

is an identifier of the nickname that you specified. The maximum length of a tag is 255. For more information on tags, see the NAMEFIND command.

Value:

is the value for the corresponding tag of the specified nickname. Values may have a maximum length of 255 characters.

Note: Use caution when entering a colon (:) in a tag field or tag value field. Colons are delimiters and if you enter them incorrectly it may cause unpredictable results.

3. Local User ID:

Some CMS commands, such as GRANT AUTHORITY, need a local user ID to identify users on other computer systems. These local user IDs are needed to access resources in a TSAF (Transparent Services Access Facility) collection.

When you are creating a nickname for a user on another system, specify the user ID and node ID for use by programs (such as RSCS) that use the node ID. In addition, specify a local user ID by entering a tag in the optional section at the bottom half of the display. The tag is "LOCALID." If the user is on a system that is part of your TSAF collection, the value for the LOCALID tag is the same as the user's *userid*. If the user is on a system that is not part of your TSAF collection, you will have to ask them for the local user ID that was assigned to them in your TSAF collection. Enter this local user ID for the value of the LOCALID tag. This is an example of how to specify a local user ID for a user on a system in another TSAF collection:

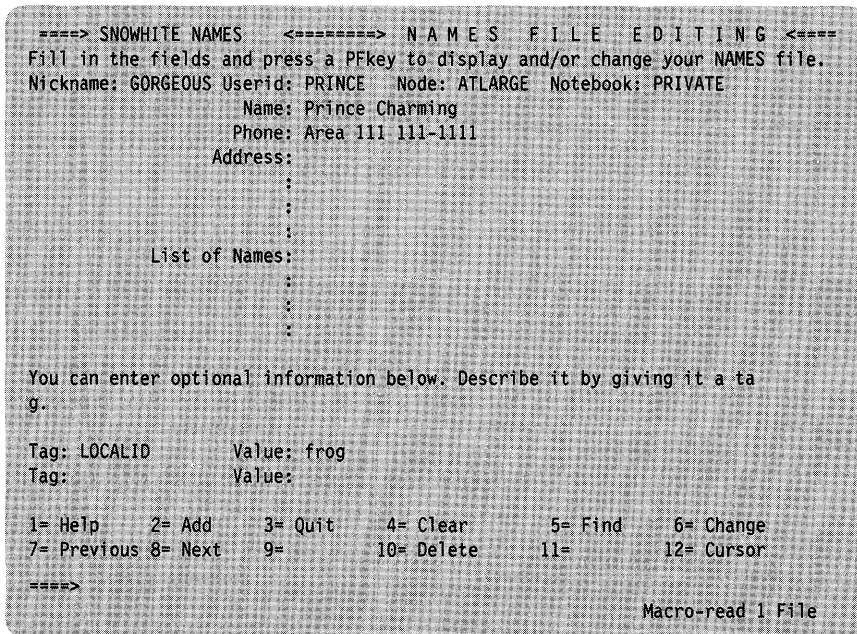


Figure 13. Sample Entry for a Local User ID.

The value you enter for the LOCALID tag cannot be a nickname and it cannot contain a node ID. The value can be a list of local user IDs if you want to use a nickname to specify a group of users on another system.

If you are setting up a nickname for a user on your system, no local user ID is necessary.

4. PF Key Settings on the NAMES Menu

The PF key functions appear on the NAMES menu itself (see “Examples”) and are summarized in the following list:

PF 1	Help	Display NAMES command description.
PF 2	Add	Add this entry to the NAMES file.
PF 3	Quit	Exit from menu.
PF 4	Clear	Clear input fields.
PF 5	Find	Locate in the file the first field that is filled in on the menu.
PF 6	Change	Change this entry.
PF 7	Previous	Display the previous entry.
PF 8	Next	Display the next entry.
PF 9		Not assigned.
PF 10	Delete	Delete this entry.
PF 11		Not assigned.
PF 12	Cursor	If cursor is on the menu, move it to the command line; if cursor is on the command line, move it back to its previous location on the menu.

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here. On a display terminal without PF keys, you can enter QUIT on the command line to exit from the screen.

Pressing the PA1 key while in the NAMES menu displays the WM window, unless the CP TERMINAL BRKKEY has been assigned to PA1.

5. Updating a *userid* NAMES File

You can make changes to the file by using the menu and appropriate PF keys (see above), or by editing the file (XEDIT *userid* NAMES). If you issue NAMES from a line mode terminal, you are placed in edit mode, editing the file *userid* NAMES. The format of a *userid* NAMES file is shown in the “Examples” section of the NAMEFIND command.

6. If you want to issue NAMES from an exec program, you should precede it with the EXEC command; that is, specify

```
exec names
```

Note: When you are working with a names file, and the NODE tag in the NAMES file indicates that the user is on another processor, then you must also specify a LOCALID tag.

Examples

The following is an entry in the file SNOWWHITE NAMES.

```

====> SNOWHITE NAMES <=====> N A M E S   F I L E   E D I T I N G <====
Fill in the fields and press a PFkey to display and/or change your NAMES file.
Nickname: SNOW   Userid: SNOWHITE Node: FOREST   Notebook:
      Name: Snow White
      Phone: ZZZ-ZZZZ
      Address: Forest Primeval
      :
      :
      :
      List of Names:
      :
      :
      :

You can enter optional information below. Describe it by giving it a tag.

Tag:           Value:
Tag:           Value:

1= Help   2= Add   3= Quit   4= Clear   5= Find   6= Change
7= Previous 8= Next   9=       10= Delete  11=      12= Cursor

====>
Macro-read 1 File

```

Figure 14. Sample NAMES Screen

The following menu shows an entry for a list of names. Each name in the list is the nickname of an entry in the names file.

```

====> SNOWHITE NAMES <=====> N A M E S   F I L E   E D I T I N G <====
Fill in the fields and press a PFkey to display and/or change your NAMES file.
Nickname: DWARFS   Userid:           Node:           Notebook:
      Name:
      Phone:
      Address:
      :
      :
      :
      List of Names: SNOOZY DUMMY BOSS SMILEY GROUCHY SNIFFLES WISTFUL
      :
      :
      :

You can enter optional information below. Describe it by giving
it a tag.

Tag:           Value:
Tag:           Value:

1= Help   2= Add   3= Quit   4= Clear   5= Find   6= Change
7= Previous 8= Next   9=       10= Delete  11=      12= Cursor

====>
Macro-read 1 File

```

Figure 15. Sample Entry for a List of names.

Responses

name has been added to your *userid* NAMES file.
 Entry has been deleted from your *userid* NAMES file.
 Entry changed in your *userid* NAMES file.
 Warning: There {is|are} *nn* undisplayed tag(s).

The following response is displayed on a line mode terminal:
 You are now editing your Userid NAMES File.

Messages and Return Codes

DMSWNM006E No read/write filemode accessed [RC=36]
 DMSWNM653E Error executing GLOBALV, rc=*nn* [RC=*nn*]

Messages when in the NAMES panel

DMSWNM645W The user tag name *name* is too long to display in the panel
 DMSWNM656E Error searching your NAMES file; rc=*nn* from NAMEFIND command [RC=100]
 DMSWNM657E Undefined PFkey/PAkey
 DMSWNM658W The value for the *tag* tag is too long to display on the panel
 DMSWNM660E The nickname field must be filled in
 DMSWNM660W Warning: this entry duplicates an existing nickname
 DMSWNM662E You are not on an entry; press PF 5, 7 or 8 to move to an entry
 DMSWNM663W There is/are *nn* undisplayed tag(s)
 DMSWNM664E Entry not found
 DMSWNM664E Next entry not found
 DMSWNM664E Previous entry not found

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

NETDATA

Use the NETDATA command from within an exec to query, receive, or send files, notes, or acknowledgements that are in the NETDATA format. Although it is not recommended and is certainly more complex, you may use NETDATA from the command line.

The NETDATA command is invoked when you issue the NOTE, PEEK, RECEIVE, and SENDFILE commands, or when you issue the DISCARD subcommand from your reader. It is neither designed nor intended for use from the command line, although it is possible, but inconvenient to use it in such a way. The most frequent invocation of NETDATA is normally from inside an exec.

Format

NETDATA	<pre> { QUERY [(optionA... [])] RECEIVE <i>fn ft fm</i> [(optionA optionB optionD... [])] SEND <i>fn ft fm</i> TO <i>userid</i> AT <i>node</i> [(optionA optionC optionD... [])] } </pre> <p>OptionA: [<u>TYPe</u>] [<u>MSGSubs</u>] [<u>NOType</u>] [<u>MSGAll</u>] [<u>STACK</u> [<u>FIFO</u>]] [<u>LIFO</u>] [<u>LIFO</u>]] [<u>FIFO</u>]]</p> <p>OptionB: [<u>Fullprompt</u>] [<u>NEwdate</u>] [<u>Replace</u>] [<u>Minprompt</u>] [<u>Olddate</u>] [<u>NOReplace</u>] [<u>NOPrompt</u>]] [<u>NOTE</u>] [<u>NOTEBook</u> <i>fn</i>] [<u>PURGE</u>]]</p> <p>OptionC: [<u>Ack</u>] [<u>NOTE</u>] [<u>NOAck</u>]]</p> <p>OptionD: [<u>Log</u>] [<u>NOSpool</u>] [<u>Xedit</u>] [<u>NOLog</u>]]</p>
----------------	--

Operands**QUERY**

requests information about the current reader spool file, if that file is in NETDATA format.

RECEIVE *fn ft fm*

requests that the current reader spool file be processed, if that file is in NETDATA format. The *fn ft fm* specifies the file identifier to be given to the incoming file. An equal sign (=) can be used for any part of the file identifier to indicate that the file name, file type, or file mode is to be the same as that of the file in the spool file.

SEND *fn ft fm*

requests that a note or a file be transmitted to a user ID at a network node in NETDATA format. The *fn ft fm* specifies the file identifier of the file to be sent. An asterisk (*) can be used for the file mode to indicate that all accessed disks or directories be searched for the file.

TO *userid*

specifies the user on your system or another system to whom the file is to be sent.

Note: This operand must be a user ID, not a nickname in your *userid* NAMES file.

AT *node*

specifies the node to which the file is to be sent.

Note: This operand cannot be omitted, even if the user to whom the file is to be sent is on the same system as you are.

Options

Duplicate and conflicting options are allowed and do not result in diagnostic messages. The rightmost option overrides any conflicting options previously entered.

Options A:**TYPE**

specifies that messages, whether informational or error, are to be typed at the user's terminal. The TYPE option overrides the FIFO, LIFO, NOTYPE, and STACK options. This is the default.

NOType

specifies that messages, whether informational or error, are not to be typed. The NOTYPE option overrides the FIFO, LIFO, STACK, and TYPE options.

STACK [FIFO]**STACK LIFO**

specifies that messages, whether informational or error, are to be placed in the stack in the order in which they would have been typed. The STACK and STACK FIFO option override the LIFO and TYPE options and force the NOTYPE option.

Note: STACK, STACK FIFO, and FIFO are synonymous. STACK LIFO, and LIFO are synonymous.

LIFO

specifies that messages, whether informational or error, are placed in the stack in the inverse order in which they are typed. The LIFO option overrides the FIFO, STACK, and TYPE options and forces the NOTYPE option.

FIFO

specifies that messages, whether informational or error, are placed in the stack in the order in which they are typed. The FIFO option overrides the LIFO and TYPE options and forces the NOTYPE option.

MSGSubs

returns only the available substitution information for the current spool file. Substitution data in a message is the variable information contained in the message. See the usage notes for additional information on substitution data.

The lines are displayed or stacked in accordance with the NOTYPE, STACK, LIFO, or FIFO options.

MSGAll

returns the normal message and all available substitution information for the current spool file. The lines are displayed or stacked in accordance with the NOTYPE, STACK, LIFO, and FIFO options described above.

Options B:**Fullprompt**

specifies that a prompt is to be issued for the incoming file. The FULLPROMPT option overrides the MINPROMPT, NOPROMPT, NOSPOOL, and XEDIT options.

Minprompt

specifies that a prompt is to be issued for the incoming file whenever its name is different from that of the spool file. The MINPROMPT option overrides the FULLPROMPT, NOPROMPT, NOSPOOL, and XEDIT options. This is the default.

NOPrompt

specifies that no prompt is to be issued for the incoming file. The NOPROMPT option overrides the FULLPROMPT and MINPROMPT options.

NEwdate

specifies that the date and time recorded for the incoming file is that when it was received. The NEWDATE option overrides the OLDDATE option.

Olddate

specifies that the date and time recorded for the incoming file is the date when it was created or last updated by the sender. The OLDDATE option overrides the NEWDATE option. This is the default.

Replace

specifies that an existing file is to be replaced by the incoming file. The REPLACE option overrides the NOREPLACE option.

NOREplace

specifies that an existing file is not to be replaced by the incoming file. The NOREPLACE option overrides the REPLACE option. This is the default.

NOTE

specifies that the file is to be sent or received as a note.

NOTEBook *fn*

specifies the name of the NOTEBOOK file to which the note being received is to be appended, following a line of 73 equal signs (=). This option is ignored if the file being received is not a note or if the NOTE option was not specified.

PURGE

specifies that the file being received is to be purged, not received.

Options C:**Ack**

specifies that an acknowledgement is to be returned to your reader when the recipient receives or discards the file. The ACK option overrides the NOACK option.

NOAck

specifies that no acknowledgement is to be returned to you when the recipient receives or discards the file. The NOACK option overrides the ACK option. This is the default.

NOTE

specifies that the file is to be sent or received as a note.

Options D:**Log**

specifies that the SEND or RECEIVE operation is to be logged in your *userid* NETLOG file. Log entries are described in a following section. The LOG option is forced when receiving an acknowledgement. The LOG option overrides the NOLOG option.

NOLog

specifies that the SEND or RECEIVE operation is not to be logged in your *userid* NETLOG file. The NOLOG option is forced when purging an acknowledgement and overrides the LOG option. This is the default.

NOSpool

specifies that the status of the punch is not altered during a SEND operation. NOSPOOL also specifies that the status of the reader is not altered during a RECEIVE operation. During processing, no CP TAG or CP SPOOL commands are issued. Acknowledgement files will be sent during RECEIVE operations and the punch will be saved, spooled, and tagged, and then restored to send the acknowledgement file. The NOSPOOL operation overrides the FULLPROMPT and MINPROMPT options and forces the NOPROMPT option.

Xedit

specifies that the file being sent should be written from XEDIT storage instead of from a minidisk or directory or that the file being received should be read into XEDIT storage instead of to a minidisk or directory. The XEDIT option overrides the FULLPROMPT and MINPROMPT options and forces the NOPROMPT option.

Usage Notes**1. Tailoring the NETDATA Command Options**

You can use the DEFAULTS command to set up options that override the command defaults for the NETDATA RECEIVE and NETDATA SEND commands. However, the options you specify in the command line when entering either of these commands override those specified in the DEFAULTS command. This allows you to customize the defaults for these commands, yet override them when desirable. Refer to the DEFAULTS command for more information.

2. Acknowledgements

Acknowledgements can be sent to users on different computer systems connected by the RSCS network so that they can be sure that a file they sent was received.

The sender can specify on the NETDATA SEND command that an acknowledgement be returned when a file is received. Even if a recipient discards a file, an acknowledgement is returned to the sender. The acknowledgement indicates whether the file was received (written to a disk or directory) or discarded (purged).

When you receive an acknowledgement that appears in your reader, all parameters and all options (except the *spoolid* and the PURGE option) are ignored. The acknowledgement is used to make an entry in your *userid* NETLOG file. This entry confirms that the file you sent was received (or discarded). The format of entries in the *userid* NETLOG file is shown in the Examples section below.

3. The NETDATA RECEIVE command resets the continuous spooling option and spools your reader NOCONT unless the NOSPOOL option is specified.

4. Substitution data in a message is the variable information contained in the message. For example, *fn ft fm*.

5. The substitution data line generated by MSGSUBS and MSGALL options contains the message identifier of the actual message, followed by any substitution data. Substitution data is returned only on zero return codes, with one exception; return code 32 (DMSDDL636W Received null file; no file created). The NETDATA command can receive a null or empty file. However, CMS does not support empty files; therefore, the file is not allowed to exist.

Return code 32 is a warning of a CMS null file condition. NETDATA received this file and you received the requested substitution data as though the return code was zero.

6. If NETDATA is issued with the MSGSUBS or MSGALL option and the TYPE, NOTYPE, STACK, FIFO, or LIFO options are not specified, then the result is returned to the terminal.

7. If MSGSUBS is specified, only the message identifier and the available substitution data is returned. The message itself is not stacked or returned to the terminal.

8. If MSGALL is specified, both the actual message and the message identifier with the available substitution data are returned. The actual message is the first line returned, and the substitution data is the second line returned.

9. Responding to Prompting Messages

If you specify the FULLPROMPT or MINPROMPT option on a NETDATA RECEIVE command, the valid responses include:

- One of the digits specified in the prompt
- One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	If this file is one of a set of files that constitutes a single spool file, the file is not received and prompting continues for the next file, if there is one. If this is the last file of a set of files or if this is the only file in the spool file, the command is ended.
1 or Yes	Receives the file under the name <i>fn1 ft1 fm1</i> (or <i>fn3 ft3 fm3</i>). See the Responses for a definition of <i>fn1 ft1 fm1</i> and <i>fn3 ft3 fm3</i> .
2 or Quit	Ends the command.
3 or Rename	Requests prompt message DMSDDL1080R so the incoming file can be received using a different name.

10. If you receive prompt message DMSDDL1081R, the valid responses include

- One of the digits specified in the prompt
- One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	Does not receive the file under the name <i>fn fn fm</i> and repeats the original prompt message DMSDDL1079R. This allows you to specify a different name for the incoming file.
1 or Yes	Receives the file under the name <i>fn ft fm</i> .
2 or Quit	Ends the command.

11. Which prompt is most useful for you?

- If you do not issue either a QUERY RDR command or a RDRLIST command, specify FULLPROMPT.
- If you do issue a QUERY RDR command before issuing the NETDATA RECEIVE command, specify MINPROMPT.
- If you issue the NETDATA RECEIVE command from a controlled environment (such as RDRLIST) where the identities of all incoming files are known, specify NOPROMPT.

12. Special NETDATA Files from MVS with TSO Extensions (PP)

The MVS with TSO Extensions Program Product can send an empty file, in which case NETDATA RECEIVE will give you an error message indicating that no file was created. It can also send, as a unique case of multiple files in one transmission, one note and a data file together. The note will be the first file in the transmission and the data file will be second. NETDATA RECEIVE will add the note to the appropriate notebook, receive the data file, and give informative messages for each action. This is the only form of multiple NETDATA files supported by the NETDATA RECEIVE command.

Note: NETDATA RECEIVE will not handle partitioned data sets or data sets that have been encrypted by Access Method Services (AMS). These files will not be received and remain in the reader. Multiple NETDATA transmissions that do not have a note as the first file result in the first file being received and the other file(s) ignored. The entire spool file is left in the user's reader.

13. Receiving NETDATA Files Using the OLDDATE Option

If a file is sent in NETDATA format from one location to another in a different time zone and it is received using the OLDDATE option of the NETDATA RECEIVE command, the date and time of the file reflect when it was last modified relative to GMT (Greenwich Mean Time).

For example, suppose a file was last modified on 1 January 1986 at 14:00 in a time zone that is eight hours west of GMT and it is sent to a time zone five hours west of GMT. When the file is received, the time and date are changed to 1 January 1986 at 17:00.

14. Priority

When the NETDATA SEND command is used to send a note or a file across the network (to a node different from yours), the file is assigned a priority. The order and speed of transmission are based on both this priority and the size of the file. See the *Application Development Reference for CMS* for a complete description of assigned priorities.

15. NETDATA SEND always sends files as CLASS A NOCONT NOHOLD, regardless of the class to which you spool your PUNCH. The CP message that is generated, containing the *spoolid*, etc., is suppressed.

16. The format of the data transmitted and received by the NETDATA command is described in the *VM/SP CMS Diagnosis Reference*.

Example

1. Entries in a Sender's NETLOG File:

```
File SMALL  DATA  A1 sent  to MAUREEN  at BROOKLYN on 06/17/88 18:04:14
Note OHARA  NOTE   A0 sent  to MAUREEN  at BROOKLYN on 06/17/88 18:15:26
Ackn 06/17/88 18:05:41  recv  by MAUREEN  at BROOKLYN on 06/17/88 18:04:14
Ackn 06/17/88 18:15:41  recv  by MAUREEN  at BROOKLYN on 06/17/88 18:15:26
Note OHARA  NOTE   A0 sent  to MAUREEN  at BROOKLYN on 06/26/88 11:05:47
Ackn 06/26/88 11:14:05  disc  by MAUREEN  at BROOKLYN on 06/26/88 11:05:47
```

where:

recv means received. The file ID may be received as a different file ID than when it was sent.

disc means discarded.

2. Entries in a Recipient's NETLOG File:

```
File NEW      DATA  A1 recv from OHARA   at CAMBRIDG on 06/17/88 18:05:26
sent as SMALL DATA A1
Note OHARA   NOTE   A0 recv from OHARA   at CAMBRIDG on 06/17/88 18:15:36
Note OHARA   NOTE   A0 disc from OHARA   at CAMBRIDG on 06/26/88 11:12:00
```

Responses

1. If you specify the FULLPROMPT or MINPROMPT option, one of these prompts is displayed:

DMSDDL1079R Receive *fn1 ft1 fm1*?

Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* and replace the existing file of the same name?

Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* and replace *fn2 ft2 fm2*?

Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3*?

Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and replace the existing file of the same name?

Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3*

and replace *fn2 ft2 fm2*?

Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

- The file ID *fn1 ft1 fm1* is the name from the NETDATA records in the spool file.
 - The phrase “and replace the existing file of the same name” appears when the operation replaces an existing file and the file mode of that file is the same as *fm1*.
 - The phrase “and replace *fn2 ft2 fm2*” appears when the operation replaces an existing file and the file mode of that file is not *fm1*.
 - The file ID *fn3 ft3 fm3* is the name you have specified in response to prompting message DMSDDL1080R.
2. If you respond with a 3 (or RENAME) to prompting message DMSDDL1079R, you will receive the prompting message
- DMSDDL1080R Enter the new name for *fn ft fm*
and you must enter a file ID of the form *fn [ft [fm]]*.
3. If you respond to prompting message DMSDDL1080R with a file ID that names an existing file, you will receive the prompting message
- DMSDDL1081R Replace *fn ft fm*?
Reply 0 (NO), 1 (YES), or 2 (QUIT)
4. Successfully receiving a spool file will cause one of the following responses to be issued:
- If the incoming file (*fn1 ft1 fm1*) does not already exist:
File *fn2 ft2 fm2* created from *fn1 ft1 fm1* received from *userid* at *node*
 - If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*):
File *fn2 ft2 fm2* replaced by *fn1 ft1 fm1* received from *userid* at *node*
 - If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*), but is given a mode (*fm3*) that differs from the mode of the existing file (*fm2*):
File *fn3 ft3 fm3* replaced *fn2 ft2 fm2* with *fn1 ft1 fm1* received from *userid* at *node*

5. Other responses include:

- File *spfn spft* has been discarded
- Note *spfn spft* has been discarded
- Note *spfn spft* added to *fn* NOTEBOOK *fm*
- Ackn *date time* added to *userid* NETLOG
- Ackn *date time* has been discarded

6. If you specify the MSGSUBS or MSGALL option, the NETDATA command uses the following response format when returning the substitution data. This response has a fixed format and fixed length. The length, including blank delimiters, is 183 characters.

Notes:

- a. Each individual field in the substitution line is initialized to an "*" followed by enough blanks to fill the field, except the LRECL field which is numeric and initialized to zero.
- b. Initialization to the full length of the variables is done to satisfy assembler programmers who call NETDATA and require the full length of each field to avoid parsing.

The format is (each field is delimited by one blank):

```
msgid fileid noteid userid nodeid logdate1 logtime1 logdate2 logtime2 lrecl overlay sentname
```

where:*msgid*

consists of a four-digit message number concatenated with a two-digit format number. It is the actual message number of the message issued at the completion of the NETDATA command and has a length of six.

fileid

consists of three fields, each delimited by one blank: file name (*fn*), length of eight; file type (*ft*), length of eight; and file mode (*fm*), length of two.

- NETDATA RECEIVE or NETDATA QUERY of a file specifies the *fn*, *ft*, and *fm* that the file will be received as.
- NETDATA RECEIVE or NETDATA QUERY of a note specifies the *fn*, *ft*, and *fm* that will contain the note.
- NETDATA RECEIVE or NETDATA QUERY of an acknowledgement specifies the *fn*, *ft*, and *fm* of the netlog that will contain the acknowledgement.
- NETDATA SEND of a file or note specifies the *fn*, *ft*, and *fm* of the file or note that is being sent.

noteid

consists of three fields, each delimited by one blank: file name (*fn*), length of eight; file type (*ft*), length of eight; and file mode (*fm*), length of two.

- NETDATA RECEIVE or NETDATA QUERY of a note specifies the *fn*, *ft*, and *fm* of the note that was sent to you.

- NETDATA RECEIVE or NETDATA QUERY of a file or an acknowledgement will not set the *noteid*; therefore, it specifies the initialized value of '*' in each of the three fields.
- NETDATA SEND of a file or a note will not set the *noteid*; therefore, it specifies the initialized value of '*' in each of the three fields.

userid

specifies a user ID, length of eight, for the following:

- NETDATA RECEIVE or NETDATA QUERY of a file, a note, or an acknowledgement specifies the user ID of the user who sent the file, note, or acknowledgement.
- NETDATA SEND of a file or a note specifies the user ID you are sending the file to.

nodeid

specifies a node ID, length of eight, for the following:

- NETDATA RECEIVE or NETDATA QUERY of a file, a note or an acknowledgement specifies the node ID of the user who sent the file, note, or acknowledgement.
- NETDATA SEND of a FILE or a NOTE specifies the node ID you are sending the file to.

logdate1

specifies the date, length of eight, when the following was issued:

- NETDATA RECEIVE, NETDATA QUERY, or NETDATA SEND for a file or note.
- NETDATA RECEIVE, NETDATA QUERY, or NETDATA SEND for an acknowledgement is the date the original file or note was sent.

logtime1

specifies the time, length of eight, when the following was issued:

- NETDATA RECEIVE, NETDATA QUERY, or NETDATA SEND for a file or note.
- NETDATA RECEIVE, NETDATA QUERY, or NETDATA SEND for an acknowledgement is the time the original file or note was sent.

logdate2

specifies the date, length of eight, when you issued the following:

- NETDATA RECEIVE or NETDATA QUERY for an acknowledgement.
- NETDATA RECEIVE, NETDATA QUERY, or NETDATA SEND for a file or a note will not set the date; therefore, it specifies the initialized value of '*'.

logtime2

specifies the time, length of eight, when you issued the following:

- NETDATA RECEIVE or NETDATA QUERY for an acknowledgement
- NETDATA RECEIVE, NETDATA QUERY, or NETDATA SEND for a file or a note will not set the time; therefore, it specifies the initialized value of '*'.

lrecl

specifies the logical record length, length of 14, for the following:

- NETDATA RECEIVE or NETDATA QUERY of a file or a note.
- NETDATA SEND of a FILE or a NOTE will not set the logical record length; therefore, it specifies the initialized value of zero.
- NETDATA RECEIVE or NETDATA QUERY of an acknowledgement will not set the logical record length; therefore, it specifies the initialized value of zero.

overlay

consists of three fields, each delimited by one blank: file name (*fn*), length of eight; file type (*ft*), length of eight; and file mode (*fm*), length of two.

- NETDATA RECEIVE specifies the *fn*, *ft*, and *fm* of the file you are replacing. If the REPLACE option is not specified, this field is initialized to a value of "*" in each of the three fields.

Note: It is used when the file you are replacing has the same *fn* and *ft*, but has a different *fm*.

For example: If TEST FILE A1 exists and you issue the following command:

```
NETDATA RECEIVE TEST FILE A2 (MSGALL REPLACE
```

overlay will be TEST FILE A1.

- NETDATA QUERY or NETDATA SEND is not set for this field; therefore, it specifies the initialized value of "*" in each of the three fields.

sentname

specifies the name of a file for the following:

- NETDATA RECEIVE or NETDATA SEND specifies the name of the file as specified by the sender. The value in this field is dependent on the origin of the file. If the file originated on a CMS system it is in the form *fn ft fm*. If it originated on the MVS system, it is in the standard MVS file format (*xxxxxx.xxxx.xxxx*__ up to 44 characters in length).
- NETDATA SEND is not set for this field; therefore, it specifies the initialized value of "*".

Messages and Return Codes

DMSDDL006E	No read/write filemode accessed for <i>fn ft</i>
DMSDDL024E	File <i>fn ft fm</i> already exists; specify REPLACE option [RC=28]
DMSDDL037E	Filemode <i>mode</i> accessed as read/only
DMSDDL069E	Filemode <i>mode</i> not accessed
DMSDDL078E	Invalid card in reader deck [RC=32]
DMSDDL109S	Virtual storage capacity exceeded [RC=104]
DMSDDL257T	Internal system error at address <i>hex</i> (offset <i>hex</i>)
DMSDDL636E	Unsupported type of NETDATA file [RC=88]
DMSDDL636W	Received null file; no file created [RC=32]
DMSDDL638E	<i>entry type</i> is too wide to append to <i>fn ft</i> [RC=32]
DMSDDL639E	Error in <i>name</i> routine; return code was <i>nnnnnnnn</i> [RC=100]
DMSDDL688E	XEDIT option only valid from XEDIT environment [RC=24]
DMSDDL1123E	Unknown response <i>text</i> ignored

DMSDDL1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC = 1]

DMSDDL1138E Filesharing conflict involving file *fn ft fm* Error *nnn* opening file *fn ft fm* [RC = 31|55|99|100]

DMSDDL1262S Error *nnn* closing file *fn ft fm* [RC = 31|100]

DMSDDL1285S Default option *option* is invalid [RC = 24]

DMSXCM514E Return code *nn* from NETDATA [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

NOTE

Use the NOTE command to prepare a “note” for one or more computer users on your computer or on other computers connected to yours via the Remote Spooling Communications Subsystem (RSCS) network. A “note” is a short communication, the kind usually done by letter. Some of the features of the NOTE command are:

- The System Product Editor (XEDIT) controls the environment in which a note is prepared. Therefore, the full power of the editor is available to help you prepare notes.
- NOTE is one of several commands that references a “userid NAMES” file. By setting up a names file, you can identify recipients just by using nicknames, which are automatically converted into node and user ID. For information on creating a names file, see the NAMES command.
- Notes can be sent not only to individual users but also to everyone on a list.
- Headings identifying the sender and the recipients are automatically generated on each note. The information in the headings is collected from the “userid NAMES” file. Notes can be prepared with either short or long headings. An example of each is shown in the “Examples” section, below.
- PF keys are assigned to frequently used functions like sending the note, tabbing, calling for HELP, etc.

Format

NOTE	<p style="text-align: center;">[<i>name...</i> [CC: <i>name...</i>]] [(options... [])]</p> <p>Options:</p> <table style="width: 100%; border: none;"> <tr> <td style="border: 1px solid black; padding: 2px; text-align: center;">ACK</td> <td style="border: none; padding: 0 10px;">[ADd]</td> <td style="border: none; padding: 0 10px;">[Cancel]</td> <td style="border: 1px solid black; padding: 2px; text-align: center;">NOTEbook <i>fn</i></td> </tr> <tr> <td style="border: 1px solid black; padding: 2px; text-align: center;">NOAck</td> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: 1px solid black; padding: 2px; text-align: center;">NOTEbook *</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px; text-align: center;">NONotebook</td> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> </table> <table style="width: 100%; border: none;"> <tr> <td style="border: 1px solid black; padding: 2px; text-align: center;">LOG</td> <td style="border: none; padding: 0 10px;">[LONg]</td> <td style="border: none; padding: 0 10px;">[Replace]</td> <td style="border: 1px solid black; padding: 2px; text-align: center;">PROFile <i>fn</i></td> </tr> <tr> <td style="border: 1px solid black; padding: 2px; text-align: center;">NOLog</td> <td style="border: none; padding: 0 10px;">[ShorT]</td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> </table>	ACK	[ADd]	[Cancel]	NOTEbook <i>fn</i>	NOAck			NOTEbook *	NONotebook				LOG	[LONg]	[Replace]	PROFile <i>fn</i>	NOLog	[ShorT]		
ACK	[ADd]	[Cancel]	NOTEbook <i>fn</i>																		
NOAck			NOTEbook *																		
NONotebook																					
LOG	[LONg]	[Replace]	PROFile <i>fn</i>																		
NOLog	[ShorT]																				

Operands*name*

is one or more “names” of the computer users to whom the note is to be sent. If the same recipient is specified more than once, he receives only one copy of the note. The “name” may take any of the following forms, and the different forms can be freely intermixed:

- a “nickname” that can be found in the file “userid NAMES,” where “userid” is your user ID. This nickname may represent a single person (on your computer or on another computer), or a list of several people.

- a user ID of a user on your computer. If a name cannot be found in the “userid NAMES” file, it is assumed to be a user ID of a user on your computer.
- “userid AT node,” which identifies a user (“userid”) on your computer or another computer (“node”).

A user ID cannot be “AT” or “CC:.”

CC:

indicates the following name(s) are “complimentary copy” recipients of the note. A name can take any of the forms described above. Complimentary copy recipients are designated as such in the note header.

Issued without parameters, NOTE is used to continue a note that was started previously. For more information on saving and continuing notes, see the usage note, “Continuing Notes.”

Options**ACk**

requests an acknowledgment be sent to you when the addressee receives your note. For more information on acknowledgments, see the RECEIVE command description.

NOAck

requests that no acknowledgment be sent. This is the default.

ADd

causes the addressees to be added to the current invocation of NOTE. No other options may be specified when the ADD option is used. This option is intended to be used from within the NOTE command environment. For more information on this option, see the usage note, “Adding and Deleting Names of Recipients.”

Cancel

causes the note you are currently editing to be erased. You are returned to the file you were previously editing or to CMS, and no note is sent. You enter NOTE with the CANCEL option from the XEDIT command line. All other options are ignored if CANCEL is specified.

NOTEbook *fn*

causes the text of the outgoing note to be saved in a file named “fn NOTEBOOK.” You can use this option if you want a copy of the note(s) sent to a particular recipient to be kept in a separate file.

If you do not specify a notebook file name here, a file name is first searched for in the (first) recipient’s entry in your “userid NAMES” file, and then in a file set up by the DEFAULTS command. If neither contains a notebook file name, the note is saved in the default notebook file, “ALL NOTEBOOK.” A note is saved by appending it to the NOTEBOOK file, with a separator line between each note. The separator line consists of 73 equal signs (=) with columns 74-132 reserved for IBM use.

(See the NAMEFIND or NAMES command for more information on the relationship between the NAMES file and the NOTEBOOK file.)

NOTE

NOTEbook *

specifies that the text of the outgoing note is saved in a file named "name NOTEBOOK," where "name" is the value of the Notebook tag in the recipient's entry in your "userid NAMES" file, or the recipient's nickname, or the recipient's user ID (whichever is located first).

When there is more than one recipient, the full text of the note is saved in the NOTEBOOK file of the first addressee (selected as described above). In the notebook files of the other addressees and complimentary copy recipients (if any), only the note header and a line referencing the file in which the full text exists is saved. The search order for the notebook file name for these recipients is the same as described above.

NONotebook

specifies that a copy of the outgoing note is not to be saved.

LOG

specifies that the addressees, date, and time of this note are logged in a file called "userid NETLOG," where "userid" is your user ID. This log is updated when acknowledgments are received (if they were requested). This is the default.

NOLog

specifies that this note is not to be logged.

LONg

causes the long form of the note header to be used. An example of the long form is shown in the "Examples" section, below.

Short

causes the short form of the note header to be used. An example of the short form is shown in the "Examples" section, below. This is the default.

Replace

causes the work file from a previously interrupted note to be erased before NOTE is entered. If there is no work file, this option has no effect.

PROFile *fn*

specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the NOTE command. By default the macro PROFNOTE XEDIT is invoked. For more information on the PROFNOTE macro, see the usage note, "Default PF Key Settings."

Usage Notes

1. Tailoring the NOTE Command Options

You can use the DEFAULTS command to set up options and/or override command defaults for NOTE. However, the options you specify in the command line when entering the NOTE command override those specified in the DEFAULTS command. This allows you to customize the defaults of the NOTE command, yet override them when you desire. Refer to the DEFAULTS command description for more information.

The current options for an invocation of the NOTE command are displayed as the second line of the file while the note is being prepared. You can alter some of these options (such as LOG or ACK, but not LONG or SHORT) by typing over this line. The options line is not sent with the note.

2. Composing the Note

When you enter the NOTE command, the note screen appears (with the headings). An example of a note screen is shown in the "Examples" section,

below. You type in the text of your note in the XEDIT environment. The full power of XEDIT is available while you compose your note. Initially, you are placed in edit mode (although no prefix area or scale is displayed). Text may be entered on the first available line, but it may only begin in the columns prior to the first character of the recipients' names. For example, if the recipients' names begin in column 7, the text must begin before column 7. Otherwise, you can leave the first line blank and begin your text anywhere on the next available lines. You can also enter input or power typing mode by entering the appropriate XEDIT subcommand.

The PROFNOTE macro is executed when you issue the NOTE command. It assigns values to PF keys and creates two synonyms that make the NOTE command easier to use. The synonyms are SEND and CANCEL, for "SENDFILE (NOTE" and "NOTE (CANCEL," respectively. SEND is also assigned to a PF key. (You can specify the name of a different macro in the PROFILE option if you do not want the PROFNOTE macro to be executed.)

3. Sending the Note

To send the note, you can do one of the following:

- Press the PF5 key.
- Enter SENDFILE (NOTE or SENDFILE (NOTE OLD. The OLD option should be used when the recipient does not have the RECEIVE command available to read the note. For more information on the OLD option, see the SENDFILE command.
- Enter SEND (a synonym for "SENDFILE (NOTE").

The note is sent to the addressees and is logged or saved as specified. Control is returned either to CMS or to the file that was being edited when NOTE was issued.

4. Continuing Notes

If you want to save a note and finish it later, issue the XEDIT subcommand FILE from the command line. No note is sent, but the note is kept on a disk or directory as "userid NOTE A0." To continue the note later, issue the NOTE command *with no parameters*.

5. Adding and Deleting Names of Recipients

You can add recipients to a note while composing it, that is, after you have already entered a NOTE command. To do this, issue a NOTE command with the ADD option (from the XEDIT command line), specifying the names of the additional recipient(s). For example,

```
==> NOTE name1 name2 (ADD
```

Any nicknames are resolved, and the additional recipients are automatically added to the note header.

You can also alter the address list and complimentary copy list by typing over the header lines. However, with this method, no nicknames are resolved, and no user IDs are checked for validity. Therefore, issuing the NOTE command with the ADD option is the preferred way to add recipients.

You can delete the names of recipients directly from the note screen. Just blank out the names you wish deleted from the header lines.

6. Naming Conventions for Userid and Node

NOTE

You cannot send a note to a user ID (or nickname) or node named AT or CC:, nor can your user ID be AT or CC:. Also, your user ID must contain only those characters that are valid for CMS file names.

7. Conflicting Options

If conflicting options are entered (such as ACK and NOACK) the last one entered (the rightmost) overrides the others.

8. Default PF Key Settings

The PROFNOTE XEDIT macro is executed when the NOTE command is invoked. It sets the PF keys to the following functions:

PF 1	Help	Display NOTE command description.
PF 2	Add line	Add a blank line after the line containing the cursor.
PF 3	Quit	Quit this note. The following message may be displayed: FILE HAS BEEN CHANGED. USE QQUIT TO QUIT ANYWAY.
PF 4	Tab	Tab the cursor.
PF 5	Send	Issue SENDFILE with the NOTE option.
PF 6	?	Display the last command issued.
PF 7	Backward	Scroll back one screen.
PF 8	Forward	Scroll forward one screen.
PF 9	=	Repeat the last command issued.
PF 10	Rgtright	Shift the view to the right; press again to shift back to original display.
PF 11	Spltjoin	Split a line or join two lines, at the cursor.
PF 12	Power input	Enter power typing mode (XEDIT subcommand POWERINP).

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here. If you enter the "PROFILE fn" option, the macro specified (fn XEDIT) is invoked instead of PROFNOTE XEDIT. In "fn XEDIT," you can easily change the PF key settings.

Some XEDIT subcommands are stacked by the NOTE command (for example, SET TRUNC, SET LRECL, and SET VERIFY). In order to override these settings in a profile, these SET subcommands must be stacked FIFO.

9. The format of the file created by NOTE and sent by the SENDFILE command is described in the SENDFILE command description, in the section "Format of the File Sent by SENDFILE."
10. You cannot start a new note while in NOTE.
11. Format of the Note Header Records

Header records are generated automatically in the note file. The information in the headers is collected from the defaults and options you supplied in the NOTE command.

You can change the information displayed on these lines simply by typing over them. Changing the recipients (the users listed in the "To:" line) is discussed in the usage note, "Adding and Deleting Names of Recipients."

You can also type over the NOTE command options (the "OPTIONS:" line). Because the information listed in these lines is positional, you must type over the options in the correct order.

The format of the options header record is as follows:

OPTIONS: opt1 opt2 opt3 opt4 opt5

where:

opt1 is either ACK or NOACK.

opt2 is LOG or NOLOG.

opt3 is LONG or SHORT. (This option cannot be altered.)

opt4 is NOTEBOOK or NONNOTEBOOK.

opt5 is the NOTEBOOK filename: ALL, *, or the filename specified in the NOTE command.

The other header records are:

Date: is the date and time the note is prepared.

From: is information about the sender. The format of this line depends on whether LONG or SHORT is specified.

To: is information about the recipient(s). The format of this line depends on whether LONG or SHORT is specified.

cc: is information about the complimentary copy recipient(s). The format of this line depends on whether LONG or SHORT is specified.

12. The full power of XEDIT is available to you when using the NOTE command. For example, you may want to use XEDIT subcommands to scroll through the note or file, to locate a particular word, etc.

However, some XEDIT subcommands are inappropriate in this environment. Subcommands that alter the format or heading information of "userid NOTE" (such as SET TRUNC, SET FTYPE, SET LINEND, or SET LRECL) may cause unpredictable results.

If AUTOSAVE is on and you make enough changes in the note to create an autosave file, CMS does not erase the AUTOSAVE file when you send the note. Use the ERASE command to erase the AUTOSAVE file.

13. If you want to issue NOTE from an exec program, you should precede it with the EXEC command; that is, specify:

```
exec note
```

Examples

When a NOTE command is issued, the type of heading generated depends on whether the SHORT option (the default) or LONG is specified. The short form lists only the user IDs and nodes (if different from the sender's) of the addressees. The long form also lists the name and phone number of each addressee.

An example of each type of heading is shown below. The information in the headings was collected from the names file shown in the "Examples" section of the NAMEFIND command.

The command "NOTE DWARFS CC: GORGEIOUS" where DWARFS and GORGEIOUS are nicknames in the names file referenced above, produced the following screen:

NOTE

```
SNOWHITE NOTE      A0 V 132 Trunc=132 Size=24 Line=12 Col=1 Alt=0
OPTIONS: NOACK LOG SHORT NOTEBOOK ALL

Date: 11 February 1981, 11:04:52 EDT
From: Snow White          ZZZ-ZZZZ          SNOWHITE at FOREST
To:  SNOOZY at COTTAGE, DUMMY at COTTAGE, BOSS at COTTAGE, SMILEY at COTTAGE
     GROUCHY at COTTAGE, SNIFFLES at COTTAGE, WISTFUL at COTTAGE
cc:  PRINCE at ATLARGE

Dear Guys,
  Would you be interested in hiring domestic help?
  I understand the cottage is a mess, what with your having
  to work in the mines and sing "Heigh-Ho" and all that.
  In addition to being a hard worker, my skin is white as snow,
  my lips are red as blood, and I have black hair.
                                     Sincerely,
                                     S. White

1= Help      2= Add line 3= Quit      4= Tab      5= Send      6= ?
7= Backward 8= Forward 9= =        10= Rgtleft 11= Spltjoin 12= Power input

====>

X E D I T 1 File
```

Figure 16. Sample Note with Short Headings

If the command "NOTE DWARFS CC: GORGEOUS (LONG)" is issued, the headings look like this:

```
OPTIONS: NOACK LOG LONG NOTEBOOK ALL
Date: 11 February 1981, 11:04:52 EDT
From: Snow White          ZZZ-ZZZZ          SNOWHITE at FOREST
     Forest Primeval
To:  I. M. Dozing         777-7777          SNOOZY at COTTAGE
     S. A. What           777-7777          DUMMY at COTTAGE
     T.O.P. Banana        777-7777          BOSS at COTTAGE
     H. A. Haas           777-7777          SMILEY at COTTAGE
     E. B. Scrooge        777-7777          GROUCHY at COTTAGE
     A. H. Choo           777-7777          SNIFFLES at COTTAGE
     R. U. Shy            777-7777          WISTFUL at COTTAGE
cc:  Prince Charming      111 111-1111     PRINCE at ATLARGE
```

Figure 17. Example of Long Headings

Responses

Note cancelled.

Messages and Return Codes

DMSWNT006E	No read/write filemode accessed [RC = 36]
DMSWNT399E	Too many tags or tag too long for <i>nickname</i> in <i>userid</i> NAMES file [RC = 88]
DMSWNT637E	Missing {value nodeid} for the { <i>option option operand operand</i> } [RC = 24]
DMSWNT647E	UserId not specified for <i>nickname</i> in <i>userid</i> NAMES file [RC = 32]
DMSWNT648E	UserId <i>name</i> not found; check the <i>userid</i> NAMES file [RC = 32]
DMSWNT651E	ADD must be issued from NOTE [RC = 40]
DMSWNT651E	CANCEL must be issued from NOTE [RC = 40]
DMSWNT665E	File <i>userid</i> NOTE * not found; to begin a new note, enter NOTE <i>name</i> [RC = 28]
DMSWNT666E	NOTE already exists; enter NOTE to continue or specify REPLACE option [RC = 28]
DMSWNT668E	ADD option must be specified alone [RC = 40]
DMSWNT669E	List of addressees cannot begin with CC: [RC = 24]
DMSWNT670E	No names to be added were specified [RC = 24]
DMSWNT676E	Invalid character * for Network ID [RC = 20]
DMSWNT677E	Invalid option: <i>option</i> in option line [RC = 32]

Messages when in the NOTE environment (in XEDIT)

DMSWSF579E	Records truncated to lrecl when added to fn NOTEBOOK fm [RC = 3]
DMSWNT667E	Note header does not contain the {keyword From: keyword To: OPTIONS line DATE line} [RC = 32]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

NUCXDROP

Use the NUCXDROP command to cancel nucleus extensions. If the nucleus extensions are not in a segment the storage occupied by the corresponding program is released. The NUCXDROP command uses the NUCEXT function which is described in detail in the *VM/SP Application Development Reference for CMS*.

Format

NUCXDROP	$\left\{ \begin{array}{l} \textit{name1} \\ * \end{array} \quad \left[\textit{name2...} \right] \right\}$
-----------------	--

Operands

name

Is the nucleus extension to be cancelled.

*

Means all currently loaded nucleus extensions.

Usage Notes

1. If a nucleus extension has the 'SERVICE' attribute, it is called by NUCXDROP with the following parameter list:

```

DS  0F
DC  CL8'NUCLEUS EXTENSION NAME'
DC  CL8'RESET'
DC  8X'FF'
```

The high order byte in register 1 is set to X'FF'.

2. It is the responsibility of the unloaded program to cancel any secondary nucleus extension entry points by reacting to the RESET service call issued by NUCXDROP before the main entry point is cancelled and the program unloaded. The RESET call allows programs with several entry points to cancel these at the same time, or to free static storage areas obtained from free storage.

Messages and Return Codes

DMSNXD050E Parameter missing after NUCXDROP [RC = 24]
DMSNXD070E Invalid argument *argument* [RC = 24]
DMSNXD616W *name* does not exist [RC = 28]
DMSNXD617E Error code *nn* from DMSFRET while unloading *module* module [RC = 3]
DMSNXD624W No nucleus extensions are loaded [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

NUCXLOAD

Use NUCXLOAD to install nucleus extensions. The command loads the following types of modules into free storage and installs them as nucleus extensions:

- an ADCON-free module
- a relocatable member of an OS or CMS load library
- a CMS module file that has relocation information saved. In other words, a module that was generated via the LOAD command with the RLDSAVE option, and then followed by the GENMOD command.

These modules must be serially reusable modules. The nucleus extension is invoked by issuing the name of the nucleus extension. The NUCXLOAD command uses the NUCEXT function.

Format

NUCXLOAD	$\left\{ \begin{array}{l} \textit{name} \quad [\textit{fn}] \\ \textit{name} \quad \textit{member} \quad \textit{ddname} \end{array} \right\} \left[\left(\begin{array}{l} [\text{SYstem}] \quad [\text{SErvice}] \\ [\text{ENdcmd}] \quad [\text{IMmcmd}] \quad [\text{Push}] \quad [] \end{array} \right) \right]$
----------	---

Operands

name fn

name is the name associated with this nucleus extension. The *fn* is the optional file name of a module file to be loaded and associated with *name*. The module being loaded must be an ADCON-free CMS module, a relocatable member of an OS load library, or a CMS module file that has relocation information saved. The term ADCON-free implies that the program needs no relocation, i.e., it runs correctly when loaded at an address different from that at which it was generated (via GENMOD). It allows the object module to be read directly into storage obtained from the free storage manager, after determining the size needed from the module header (or the file format, for the one-record fixed format CMS system transient routines). The term serially reusable implies that the same copy of a routine may be used by another task after the current use has been concluded. If the second argument (i.e., *fn*) is not specified, the command name is also used as the file name of the module. The relocation information in a module file is saved by using the RLDSAVE option on the LOAD or INCLUDE commands.

name

is the name to be associated with this nucleus extension.

member

must be a member of a CMS or OS load library. To create a CMS load library, see the CMS command LKED.

ddname

is the *ddname* from the FILEDEF command that must be issued prior to calling the NUCXLOAD that identifies the load library.

System

indicates that system free storage should be used, and the program is to receive control disabled, key 0. The SYSTEM option is assumed by default for transient routines generated with the SYSTEM option of the GENMOD command.

Service

indicates that service calls are accepted (for instance a PURGE from an abend).

Endcmd

indicates that the nucleus extension receives control at normal end-of-command processing.

Immcmd

indicates that this nucleus extension can be invoked as an Immediate command.

Push

causes no check to be made to see if there is already a nucleus extension of the same name. Otherwise, an existing nucleus extension is not overridden.

Usage Notes

1. Nucleus extensions remain in effect until cancelled, either explicitly or implicitly. Implicit cancellation normally occurs only for nucleus extensions of the 'user' type (during an abnormal end cleanup time when all storage of 'user' type is reclaimed). Explicit cancellation does not release the storage (if any) occupied by the nucleus extension: that is the responsibility of the program that issues the cancellation (usually the program NUCXDROP).
2. Overlay modules may not be loaded by NUCXLOAD.
3. If a module generated from a higher level language is loaded using NUCXLOAD, caution should be taken when passing parameters to the module. See the register contents in the NUCEXT macro in the *VM/SP Application Development Reference for CMS*.
4. Not all CMS macro instructions generate ADCON-free code when expanded. The macro expansion should be inspected to determine if ADCONS are present. If ADCONS are present, the program should either be loaded from a CMS load library via the NUCXLOAD command or from a CMS module file that has relocation information saved. See the LOAD or INCLUDE commands for more information about saving relocation information.
5. You cannot save the relocation information from any CMS modules that were originally loaded in the transient area and generated with the SYSTEM option.
6. The ENDCMD nucleus extensions only receive control after a command is entered from the virtual console. They do not receive control if a command is issued from an exec, a user program, or CMS SUBSET mode. All ENDCMD nucleus extensions receive control before any end-of-command cleanup is done by the CMS console command handler.
7. NUCXLOAD with the IMMCMD option allows you to give control to a nucleus extension routine whenever a specified Immediate command is invoked. These exit routines receive control as an extension of CMS I/O interrupt handling. Therefore, they receive control with a PSW key of 0 and are disabled for interrupts. The routine must not perform any I/O operations or issue any SVCs that result in I/O operations.

In addition, the exit routine must not enable itself for interrupts. DIAGNOSE instructions can be used within the exit, but the exit routine must not enable

itself for interruptions that may be caused by the DIAGNOSE (for example, DIAGNOSE X'58'). See the NUCEXT macro discussion in the *VM/SP Application Development Reference for CMS* and in the *VM/SP CMS User's Guide*.

8. Not all high level programming languages can generate serially reusable object code. For example, OS/VIS COBOL cannot be loaded as a nucleus extension because it does not generate serially reusable code.
9. All nucleus extensions that have the ENDCMD or SERVICE attributes receive control at their appropriate time regardless if they have the same name. For example, if two nucleus extensions with the same name are loaded using the PUSH option and they both have the ENDCMD attribute, they will both receive control at normal end-of-command processing.

Messages and Return Codes

DMSNXL001E	No filename specified [RC = 24]
DMSNXL070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSNXL589E	Missing FILEDEF for DDNAME SYSIN [RC = 32]
DMSNXL618E	NUCEXT failed [RC = <i>nn</i>]
DMSNXL619E	Module <i>module</i> not found [RC = 28]
DMSNXL622E	Insufficient free storage [RC = <i>rc</i>]
DMSNXL639E	Error in DMSRLD routine; return code was <i>nnn</i> [RC = 24 31 55 70 76 99]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in loading a file	322

Return Codes

- 1 Nucleus Extension already exists.
- 4 Module is marked "not executable." The module is not loaded; no nucleus extension is defined. To determine why the "not executable" flag was set, examine the information provided by the linkage editor at the time the module was created.
- 10 Module is an overlay structure. The module is not loaded; no nucleus extension is defined. Overlay structures may not be used as nucleus extensions, because CMS does not support more than one such program at a time. Only an overlay structure in the user area is supported. If this program is to be used as a nucleus extension, it must be restructured so that it does not require overlays.
- 12 Module is marked "only loadable." The module is not loaded; no nucleus extension is defined. Modules are marked "only loadable" because of an explicit command to do so at the time they are link-edited. This is typically done when a module contains data, but not executable instructions. Such a nature makes a module unsuitable for use as a nucleus extension.
- 13 The nucleus extension could not be installed.
- 24 A filename was not specified, an invalid operand was specified, or too many or extraneous operands were specified.

NUCXLOAD

- 28 This is the return code generated by NUCXLOAD when the specified module cannot be found. It is also used in the case of an error when opening a LOADLIB file, in which case message DMSSOP036E is produced by the open routine.
- 32 For NUCXLOAD, a FILEDEF command identifying the load library must be issued prior to calling NUCXLOAD.
- 41 There was not enough free storage to build the table of SCBLOCK addresses.
- 100 An unrecoverable error occurred while reading the module from a disk or directory.

NUCXMAP

Use the NUCXMAP command to get information about the currently defined nucleus extensions, including those residing in loaded saved segments. NUCXMAP displays on the console or stacks a list of the nucleus extensions. The NUCXMAP command uses the NUCEXT function.

Format

NUCXMAP	<pre> [<i>name</i>] [(options... [])] [* ALL] Options: [NOSEGMent SEGMent {<i>segname</i>} *] [ATTRibutes] [SEGInfo] [STACK [FIFO LIFO]] [ALL] FIFO LIFO </pre>
----------------	--

Operands

name

specifies the 1-8 character name of a nucleus extension. CMS will return information that describes any nucleus extension having this name. An asterisk (*) specifies that CMS display information that describes each nucleus extension.

ALL

specifies that CMS display information for:

- NUCEXT look-aside entries
- nucleus extensions.

Options

NOSEGMent

returns information on nucleus extensions that do not reside in loaded logical segments.

SEGMent *segname*

returns information on nucleus extensions that reside in the specified loaded logical segment. *segname* specifies the 1-8 character name of a logical segment. An asterisk (*) indicates that all loaded logical segments are to be searched for the specified nucleus extensions.

ATTRibutes

returns the attributes of the specified nucleus extensions. This is the default unless SEGINFO is specified.

SEGINfo

returns the segment name, if any, that contains the specified nucleus extensions.

ALL

returns information about NUCEXT look-aside entries as well as nucleus extensions.

STACK [FIFO]

STACK LIFO

specifies that the information should be placed in the program stack (for use by an exec or other program) instead of being displayed at the terminal. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked FIFO. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

specifies that the information should be placed in the program stack rather than displayed at the terminal. The information is stacked LIFO. This option is equivalent to STACK LIFO.

Usage Notes

1. You can use the NUCXMAP command to obtain information about a specific nucleus extension. For example,


```
nucxmap identify
```

 returns information about the IDENTIFY command. If there is more than one nucleus extension named IDENTIFY, CMS displays information for all of them.

You can use the ALL option to obtain information about a specific look-aside nucleus extension. For example, the command,

```
nucxmap xedit (all
```

 returns information on the XEDIT look-aside entry as well as any information on any nucleus extensions named XEDIT.
2. Because of the increased number of attributes that a nucleus extension can have in CMS, the NUCXMAP command may use more than one line to describe an entry.
3. If SEGINFO is specified, attributes will not be displayed unless ATTRIBUTES is also specified.

Examples

Entering the following command,

```
nucxmap
```

without any options will list all your nucleus extensions and their attributes.

Name	Entry	Userword	Origin	Bytes	Amode	Attributes
ENDEXEC1	003EE370	00000000	003EE370	000002F8	24	SYSTEM ENDCMD
HZ	0001B210	00000000	0001B210	00001508	ANY	SYSTEM SERVICE IMMCMD
BIGMOD	00FD2EF0	0000B848	00FD2EF0	01955050	31	

Entering,

nucxmap (seginfo

would list the segment containing the nucleus extension. For example,

Name	Entry	Userword	Origin	Bytes	Amode	Segname
ENDEXEC1	003EE370	00000000	003EE370	000002F8	24	
HZ	0001B210	00000000	0001B210	00001508	ANY	SMALLSEG
BIGMOD	00FD2EF0	0000B848	00FD2EF0	01955050	31	BIGSEG

Messages and Return Codes

DMSNXL941I Nucleus extension *name* is not loaded

DMSNXM070E Invalid parameter *parameter* [RC = 24]

DMSNXM622E Insufficient free storage [RC = *rc*]

DMSNXM624I No nucleus extensions are loaded

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

OPTION

OPTION

Use the OPTION command to change any or all of the options in effect for the DOS/VS COBOL compiler and the RPG II compiler in CMS/DOS.

Format

OPtion	<p>[options...]</p> <p><u>Options:</u></p> <table><tr><td>[DUMP NODUMP]</td><td>[DECK NODECK]</td><td>[LIST NOLIST]</td><td>[LISTX NOLISTX]</td><td>[SYM NOSYM]</td></tr><tr><td>[XREF NOXREF]</td><td>[ERRS NOERRS]</td><td>[48C 60C]</td><td>[TERM NOTERM]</td><td></td></tr></table>	[DUMP NODUMP]	[DECK NODECK]	[LIST NOLIST]	[LISTX NOLISTX]	[SYM NOSYM]	[XREF NOXREF]	[ERRS NOERRS]	[48C 60C]	[TERM NOTERM]	
[DUMP NODUMP]	[DECK NODECK]	[LIST NOLIST]	[LISTX NOLISTX]	[SYM NOSYM]							
[XREF NOXREF]	[ERRS NOERRS]	[48C 60C]	[TERM NOTERM]								

Options

If an invalid option is specified on the command line, an error message is issued for that option; all other valid options are accepted. Only those options specified are altered, and all other options remain unchanged.

DUMP

dumps the registers and the virtual partition on the virtual SYSLST device in the case of abnormal program end.

NODUMP

suppresses the DUMP option.

DECK

punches the resulting object module on the virtual SYSPCH device. If you do not issue an ASSGN command for the logical unit SYSPCH before invoking the compiler, the text deck is written to your disk or directory accessed as A.

NODECK

suppresses the DECK option.

LIST

writes the output listing of the source module on the SYSLST device.

NOLIST

suppresses the LIST option. This option overrides the XREF option as it does in DOS/VS.

LISTX

produces a procedure division map on the SYSLST device.

NOLISTX

suppresses the LISTX option.

SYM

prints a Data Division map on SYSLST.

NOSYM

suppresses the SYM option.

XREF

writes the output symbolic cross-reference list on SYSLST.

NOXREF

suppresses the XREF option.

ERRS

writes an output listing of all errors in the source program on SYSLST.

NOERRS

suppresses the ERRS option.

48C

Uses the 48-character set.

60C

Uses the 60-character set.

TERM

Writes all compiler messages to the user's terminal.

NOTERM

Suppresses the TERM option.

Usage Notes

1. If you enter the OPTION command with no options, all options are reset to their default values, that is, the default settings that are in effect when you enter the CMS/DOS environment. CMS/DOS defaults are not necessarily the same as the defaults generated on the VSE system being used and do not include additional options that are available with some DOS compilers.
2. The OPTION command has no effect on the DOS PL/I compiler nor on any of the OS language compilers in CMS.

Responses

None. To display a list of options currently in effect, use the QUERY command with the OPTION operand.

Messages and Return Codes

DMSOPT070E	Invalid parameter <i>parameter</i> [RC=24]
DMSOPT099E	CMS/DOS environment not active [RC=40]

OSRUN

Use the OSRUN command to execute a load module from a CMS LOADLIB or an OS module library. The library containing the module must have been previously identified by a GLOBAL command. For an OS module library, the library must also have been defined in a FILEDEF command. If no library has been identified by a GLOBAL command, the OSRUN command searches the \$\$SYSLIB LOADLIB library for the specified module. The CMS LOADLIB can be either a CMS disk file or a shared file in the Shared File System.

Format

OSRUN	<i>member</i> [PARM = parameters]
--------------	--

Operands*member*

is the member of a CMS LOADLIB or an OS module library to be executed.

PARM =

are the OS parameters that the user wants to pass to the module. If the parameters contain blanks or special characters, they must be enclosed in quotes. To include quotes in the parameters, use double quotes. The parameters are passed in OS format: register 1 points to a fullword containing the address of a character string headed by a halfword field containing the length of the character string. The parameters are restricted to a maximum length of 100 characters.

Note: You may not pass parameters (PARM =) to the module if you issue the OSRUN command from a CMS EXEC file. The OSRUN command can be issued from a System Product Interpreter or an EXEC 2 file with no restrictions.

Messages and Return Codes

DMSLOS002I	File(s) <i>fn</i> LOADLIB not found
DMSLOS013E	Member <i>membername</i> not found in library <i>libname</i> [RC = 32]
DMSLOS073E	Unable to open file <i>ddname</i> [RC = 28]
DMSOSR001E	No filename specified [RC = 24]
DMSOSR052E	More than 100 characters of options specified [RC = 24]
DMSOSR070E	Invalid parameter <i>parameter</i> [RC = 24]

PARSECMD

Use PARSECMD to call the parsing facility from within an exec. The CMS parsing facility parses and translates command arguments.

Format

PARSECMD	<p><i>uniqueid</i> [(options... [])]</p> <p>Options: [<u>TYPE</u>] [APPLID <i>applid</i>] [STRING <i>cmdstring</i>]</p> <p> [NOTYPE]</p>
-----------------	--

Operands*uniqueid*

is the unique identifier for a command syntax definition in a file containing Definition Language for Command Syntax (DLCS) statements. The *uniqueid* may be up to sixteen characters in length.

Options**TYPE**

displays error messages at the terminal for syntax errors that are found while parsing the exec arguments or *cmdstring*. TYPE is the default.

NOTYPE

returns the text of syntax error messages for exec arguments or *cmdstring* in a REXX or EXEC2 variable called "MESSAGE.n." MESSAGE.0 contains the number of lines in the message. Variables MESSAGE.1 to MESSAGE.n (where *n* is the value in MESSAGE.0) contain the lines of the error message text. MESSAGE.0 is 0 if there are no errors.

APPLID *applid*

is an application identifier. The *applid* must be three alphanumeric characters, and the first character must be alphabetic. For example, "DMS" is the *applid* for CMS, the default application.

STRING *cmdstring*

parses the *cmdstring* rather than the exec's arguments normally obtained from EXECCOMM. This must be the last option specified. Specify a complete command definition, including the command name, as if it were entered from the command line.

Usage Notes

1. PARSECMD returns parsing information to the exec in a series of REXX (or EXEC 2) variables in the form

token.n and *code.n*

where *n* is a subscript that distinguishes the different values returned. The variables are in the following format:

token.0	number of tokens returned
code.0	number of validation codes returned
token.1	command name from the DLCS definition
code.1	validation code for command name
token.2	second token in the command string
code.2	validation code for the second token
.	
.	
.	
.	
token.n	nth token in the command string
code.n	validation code for the nth token

2. Possible code.n values are:

COMMAND	Command name
KEYWORD	Keyword
OPTSTART	Option start (
OPTEND	Option end)
COMMENT	Comment (everything following OPTEND)
ALPHANUM	Alphanumeric string
APPLID	Any three character alphanumeric string with first alphabetic
CHAR	A single character
CUU	Device address: X'001', X'002', ..., X'FFF'
DIGITS	Any unsigned number made up of digits 0-9
FN	Filename
FT	Filetype
EFN	Filename with '*' or '%' valid also
EFT	Filetype with '*' or '%' valid also
EXECNAME	Execname
EXECTYPE	Exectype
FM	Filemode
HEX	Hexadecimal number
INTEGER	Integer: ..., -2, -1, 0, +1, + 2, ...
NINTEGER	Negative integer: ..., -2, -1
PINTEGER	Positive integer: +1, +2, ...
MODE	Uppercase alphabetic character
STRING	Any character string (no blanks)
TEXT	Any string
usercode	A user-supplied validation code between 128 and 255 that describes the corresponding token returned by the parsing facility.

3. The TYPE and NOTYPE options apply while parsing the syntax of *cmdstring* or EXEC arguments. TYPE and NOTYPE depend on the EMSG setting in the CP SET command.

Syntax errors produced when calling PARSECMD are displayed at the terminal regardless of the TYPE and NOTYPE option.

4. The `uniqueid` you specify in the PARSECMD command is matched to the `uniqueid` specified in the DLCS file. For a detailed description of `uniqueids`, see the manual, *VM/SP Application Development Guide for CMS*.
5. PARSECMD is also the name of a CMS macro that calls the parsing facility from an assembler language program.
6. Keywords are uppercased according to the National Language Uppercase Table for the active application. If the table is not found, the CMS National Language Table is used.

Messages and Return Codes

DMSPMD407E Invalid unique ID *uniqueid* [RC=24]
 DMSPMD622E Insufficient free storage [RC=104]
 DMSPMD631E *cmdname* can only be executed from an EXEC-2 or REXX EXEC [RC=40]
 DMSPMD639E Error in *routine* routine; return code was *retcode* [RC=256]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

PEEK

Use the PEEK command to display a file that is in your virtual reader without reading it onto your disk or directory. Once you issue the PEEK command you can use XEDIT subcommands to view the file. In most cases the files in your reader were sent to you by other computer users, on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network.

Format

PEEK	<p>[<i>spoolid</i>] [(options...)]</p> <p>Options:</p> <p>[FRom <i>recno</i>] [FOR <i>numrec</i>] [PROFile <i>fn</i>]</p>
-------------	---

Operands*spoolid*

is the spoolid of the file to be displayed. The default is the “next” file in the virtual reader.

The “next” file is the one for which the RDR command returns information. Which file this is depends on the class of the reader, the class of the files in the reader, and whether or not they are held.

Options**FRom** *recno*

is the starting record number to be read. The default is 1 (one).

FOR *numrec*

is the number of records of the file to be read. Specifying an asterisk (*) causes the entire file to be used. The default is to read up to 200 records.

PROFile *fn*

specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the PEEK command. The default macro is PROFPEEK XEDIT. For more information on the PROFPEEK macro, see the usage note, “PF Key Settings on the PEEK Screen.”

Usage Notes

1. You can use the special command DISCARD from the PEEK screen. The DISCARD command allows you to purge the reader file displayed by PEEK. See “DISCARD” on page 792 for more information.
2. Tailoring the PEEK Command Options

You can use the DEFAULTS command to set up options and/or override command defaults for PEEK. However, the options you specify in the command line when entering the PEEK command override those specified in the DEFAULTS command. This allows you to customize the defaults of the PEEK

command, yet override them when you desire. Refer to the DEFAULTS command description for more information.

3. Editing from the PEEK Screen

When you invoke the PEEK command you are placed in the XEDIT environment, editing the file "spoolid PEEK A0." The full power of XEDIT is available to you while you "peek" at the file. You can make changes to this file and then issue the XEDIT subcommand FILE or SAVE from the XEDIT command line on the PEEK screen. In this case, the reader spool file is *not* changed. The changes are made only to the file that is saved or filed.

4. PF Key Settings on the PEEK Screen

The PROFPEEK macro is executed when the PEEK command is invoked, unless you specified a different macro as an option in the PEEK command. It assigns the following values to the PF keys:

PF 1	Help	Display PEEK command description.
PF 2	Add line	Add a blank line after the current line.
PF 3	Quit	Exit from the PEEK display.
PF 4	Tab	Tab the cursor.
PF 5	Clocate	Locate the string specified in an XEDIT subcommand CLOCATE or CHANGE that is typed in the command line. This PF key is set to the XEDIT macro SCHANGE 6. For more information on its use, see the publication <i>VM/SP System Product Editor Command and Macro Reference</i> .
PF 6	?/Change	Display the last command, or change the string specified in a CHANGE subcommand. (The Change function is the XEDIT SCHANGE macro and must be used in conjunction with PF5.)
PF 7	Backward	Scroll backward one screen.
PF 8	Forward	Scroll forward one screen.
PF 9	Receive	Write this file on the disk or directory accessed as A, using the same file name and file type or append the file to the appropriate notebook if the file was sent by the NOTE command.
PF 10	Rgtright	Shift the view to the right; press again to shift back to original display.
PF 11	Spltjoin	Split a line or join two lines, at the cursor.
PF 12	Cursor	If cursor is in the file area, move it to the command line; if cursor is on the command line, move it back to its previous location in the file (or to the current line).

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

If you enter the "PROFILE fn" option, the file "fn XEDIT" is invoked instead of the file PROFPEEK XEDIT. In "fn XEDIT," you can easily change the PF key settings. Some XEDIT subcommands are stacked by the FILELIST command (for example, SET TRUNC, SET LRECL, and SET VERIFY). In

order to override these settings in a profile, these SET subcommands must be stacked FIFO.

5. Files in DISK DUMP or NETDATA Format

Files in DISK DUMP or NETDATA format are reformatted so that they are readable. However, the entire file must be "peeked" at and have a logical record length of less than 256 in order to be reformatted. For more information on NETDATA format, see the SENDFILE command.

Note that if multiple files are sent with continuous spooling (using CP SPOOL PUNCH CONT) and a series of DISK DUMP commands, RECEIVE will recognize only the first file identifier (file name and file type). Any files having the same file identifier as existing files on your disk or directory accessed as A will overlay those files.

As a sender, you can avoid imposing this problem on file recipients by doing any of the following:

- a. Always use SENDFILE, which resets any continuous spooling options in effect.
- b. Do not spool the punch continuous.
- c. If you must send files with continuous spooling, warn the recipient(s) that files are being sent in this manner and list the file identifiers of the files you are sending.

Similarly, if the punch is spooled continuous and PUNCH is used to send multiple files, the file is read in as one file with ":READ" cards imbedded. In this case, although no files are overlaid, the recipient must divide the file into individual files. This problem can also be avoided by using SENDFILE or by not spooling the punch continuous.

6. Using the PEEK Command

This command is useful not only when issued in the CMS environment but also in the RDRLIST command environment. In the RDRLIST display, the PF11 key is set to the PEEK command.

7. Special NETDATA Files from MVS with TSO Extensions (PP)

The MVS with TSO Extensions program product (program number 5662-285) can send an empty file. It can also send two files in NETDATA format in a single transmission. Peeking at an empty (null) file results in a warning message that the file is empty. Peeking at two files sent in one transmission results in two messages, identifying each of the files. A line of equal signs (=) separates the two files.

8. The PEEK command does not handle MONITOR files or files with a SPECIAL status of YES. (The SPECIAL status indicates whether or not the file contains records with X'5A' carriage control characters. See the CP QUERY command to determine SPECIAL status of a file.)
9. If you want to issue PEEK from an exec program, you should precede it with the EXEC command; that is, specify
exec peek

Examples

A sample PEEK screen follows:

```

3001 PEEK A0 V 255 Trunc=255 Size=20 Line=0 Col=1 Alt=0
File NEW IDEA from OHARA at BLUESKY. Format is DISK-DUMP.
*** Top of File ***
      Small business Opportunity

Greetings...

      I am planning to open a store, in which I will sell
computer microforms and integrated circuits. I plan to call
it Bob's Fiche and Chips.

                        Bob

1= Help    2= Add line  3= Quit    4= Tab    5= Clocate  6= ?/Change
7= Backward 8= Forward  9= Receive 10= Rgleft 11= Spltjoin 12= Cursor

====>
X E D I T 1 File

```

Figure 18. Sample PEEK Screen

Responses

File *fn ft* from *userid* at *node* Format is *transmission format*

Note from *userid* at *node* Format is *transmission format*

Messages and Return Codes

DMSWPK132S File is too large [RC=88]
DMSWPK156E FROM *nnn* not found--the file *fn ft fm* has only *nnn* records
[RC=32]
DMSWPK630S Error accessing spool file [RC=36]
DMSWPK643E No class *class* files in your reader [RC=28]
DMSWPK644E All reader files are in HOLD status or not class *class* [RC=28]
DMSWPK653E Error executing EXECIO [RC=*nn*]
DMSWPK655E Spoolid *nnnn* does not exist [RC=28]
DMSWPK672E Virtual reader invalid or not defined [RC=36]
DMSWPK674E Reader is not ready [RC=36]
DMSWPK683E The entire file must be peeked at to be formatted [RC=32]
DMSWPK683W The file has an LRECL greater than 255 and cannot be
reformatted [RC=32]
DMSWPK684E File contains invalid records and cannot be reformatted [RC=32]
DMSWPK687E This is a {SYSTEM{HELD|DUMP}file|file with a special format}
and cannot be peeked [RC=1]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

	Reason	Page
	Errors in command syntax	811

PRINT

Use the PRINT command to print a CMS file with a preceding header page on the spooled virtual printer.

Note: If your virtual printer is not already defined as 00E, refer to the *Planning Guide and Reference* to set up the address.

Format

PRint	$fn\ ft\ \left[\begin{array}{c} fm \\ * \end{array} \right]\ \left[(\text{options... } []) \right]$ <p>Options:</p> $\left[\text{OVerSize} \right]\ \left[\text{CC } \left[\text{HEADer} \right] \right]\ \left[\text{UPCASE} \right]\ \left[\begin{array}{c} \text{TRC} \\ \text{NOTRC} \end{array} \right]$ $\left[\text{LINECOUN} \left\{ \begin{array}{c} nnn \\ \underline{55} \end{array} \right\} \right]\ \left[\text{MEMBER } \left\{ \begin{array}{c} * \\ \text{membername} \end{array} \right\} \right]\ \left[\text{HEX} \right]$
--------------	--

Operands

fn

is the file name of the file to be printed.

ft

is the file type of the file to be printed.

fm

is the file mode of the file to be printed. If this field is specified as an asterisk (*), the standard order of search is followed and the first file found with the given file name and file type is printed. If *fm* is not specified, the disk or directory accessed as A and its extensions are searched.

Options

OVerSize

allows you to print:

- Files that have records larger than the carriage size of the virtual printer, and
- Files that have a SPECIAL status of YES.

When the OVERSIZE option is used, the CC option will be set as a default. This default setting of CC can be overridden by specifying either the NOCC or the HEX option with the OVERSIZE option. The records that are larger than the virtual printer's carriage size are printed, but they are truncated to the carriage size. Records with a SPECIAL status of YES are printed if the record length is not greater than 32767 bytes. (The SPECIAL status indicates whether or not the file contains records with X'5A' carriage control characters. See the CP QUERY PRINTER command to determine SPECIAL status of a file.)

Note: In general, files containing the X'5A' carriage control character can only be printed on printers supported through the Print Services Facility (PSF), such as the 3820 and 3800 Model 3.

The OVERSIZE (and CC) option is assumed if the file type is LISTCPDS, LIST3820, or LIST38PP. If OVERSIZE is not specified and the file you want to print is larger than the virtual printer's carriage size, the message "Records exceeds allowable maximum" is displayed.

CC [HEADer]

interprets the first character of each record as a carriage control character. If the file type is LISTING, LIST3800, or LISTCPDS, the CC option is assumed. The first character of the first record in the file is interpreted as a page eject to ensure the file begins printing on a new page. With CC in effect, the page eject function is controlled by the carriage control characters in the file and lines per page are not counted. The LINECOUN option is ignored if CC is in effect.

HEADER creates a shortened header page with only the file name, file type, and file mode at the top of the page that follows the standard header page. The records in the file being printed begin on a new page following both header pages. The HEADER option can only be used in conjunction with the CC option. If the CC option is not specified HEADER has no effect.

NOCC

does not interpret the first character of each record as a carriage control character. In this case, the PRINT command ejects a new page and prints a heading after the number of lines specified by LINECOUN are printed. If NOCC is specified, it is in effect even if the file type is LISTING, LIST3800, LISTCPDS, LIST3820, or LIST38PP. If both NOCC and OVERSIZE are specified, then the NOCC option will override the default of CC. The maximum page value in the header is 99999. NOCC is the default.

UPcase

translates the lowercase letters in the file to uppercase for printing.

TRC

interprets the first data byte in each record as a TRC (Table Reference Character) byte. The value of the TRC byte determines which translate table the 3800 printer selects to print a record. The value of the TRC byte corresponds to the order in which you have loaded WCGMs (via the CHARS keyword of the SETPRT command). Valid values for TRC are 0, 1, 2, and 3. If an invalid value is found, a TRC byte of 0 is assumed. If the file type is LIST3800, TRC is assumed.

NOTRC

does not interpret the first data byte in each record as a TRC byte. NOTRC is the default.

LInecoun [nnn]

allows you to set the number of lines to be printed on each page. nnn can be any decimal number from 0 through 144. If a number is not specified, the default value is 55. If nnn is set to zero, the effect is that of an infinite line count and page ejection does not occur. This option has no effect if the CC option is also specified.

When calculating nnn, remember that the total number of lines printed on a page equals nnn plus 3. The 3 extra lines are for the heading (1 heading line and 2 blank lines).

MEMBER ***MEMBER** *membername*

prints the members of macro or text libraries. This option may be specified if the file is a simulated partitioned data set (file type MACLIB, TXTLIB, or LOADLIB). If an asterisk (*) is entered, all individual members of that library are printed. If a member name is specified, only that member is printed.

HEX

prints the file in graphic hexadecimal format. If HEX is specified, the options CC and UPCASE are ignored, even if specified, and even if the file type is LISTING, LIST3800, LISTCPDS, LIST3820, or LIST38PP. If both the OVERSIZE and HEX options are specified, the NOCC option will be in effect.

Usage Notes

1. The file may contain carriage control characters and may have either fixed- or variable-length records, but no record may exceed 132 characters for a 1403, 3203, or 3289 Model 4 printer, 150 characters for a 3211 printer, or 168 characters for a 4248 printer. There are exceptions:
 - If the CC option is in effect, the record length can be one character longer (133, 151, or 169) to allow for the carriage control character.
 - If the virtual printer is a 3800, you can specify a carriage control byte, a TRC byte, or both, for a total line length of up to 206 bytes.
 - If the HEX option is in effect, a record of any length can be printed, up to the CMS file system maximum of 65,535 bytes.
2. If you want the first character of each line to be interpreted as a carriage control character, you must use the CC option. When you use the CC option for files that do not contain carriage control characters, the first character of each line is stripped off. An attempt is made to interpret the first character for carriage control purposes. If the character is not valid, the results are unpredictable because CMS does not check for valid carriage control characters.

Files with a file type of UPDLOG (produced by the UPDATE command) must be printed with the CC option.
3. If the virtual printer is not a 3800 and you have specified TRC, PRINT strips off the first data byte before each line is printed.
4. One spool printer file is produced for each PRINT command; for example:


```
print mylib maclib (member get
```

prints the member GET from the file MYLIB MACLIB. If you want to print a number of files as a single file (so that you do not get output separator pages, for example), use the CP command SPOOL to spool your virtual printer with the CONT option.
5. If the MEMBER option is specified more than once, only the last member specified will be printed. However, if one MEMBER option is coded with an asterisk (*), and another MEMBER option is specified with a member name, only the specified member will be printed, regardless of their order on the command line.

For example, if you code:

```
print one maclib (member example1 member example2
```

only EXAMPLE2 will be printed. If you code:

```
print one maclib (member example1 member *
```


PRINT

only EXAMPLE1 will be printed.

6. If the printer has an extended FCB with the duplication option specified, the PRINT command is not valid because the header line is too long to be duplicated.

Responses

None.

The CMS ready message indicates the command completed without error (that is, the file is written to the spooled printer). The file is now under the control of CP spooling functions. If a CP SPOOL command option such as HOLD or COPY is in effect, you may receive a message from CP.

Messages and Return Codes

DMSVRT002E	File [<i>fn</i> [<i>ft</i> [<i>fm</i>]]] not found [RC = 28]
DMSVRT008E	Device <i>vdev</i> {invalid or nonexistent is an unsupported device type} [RC = 36]
DMSVRT013E	Member <i>membername</i> not found in library <i>libname</i> [RC = 32]
DMSVRT029E	Invalid parameter <i>parameter</i> [in the [option] <i>option</i> field] [RC = 24]
DMSVRT033E	File <i>fn ft fm</i> is not a library [RC = 32]
DMSVRT039E	No entries in library <i>fn ft fm</i> [RC = 32]
DMSVRT044E	Record exceeds allowable maximum [RC = 32]
DMSVRT069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSVRT104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]
DMSVRT109S	Insufficient free storage available [RC = 104]
DMSVRT123S	Error <i>nn</i> printing file <i>fn ft fm</i> [RC = 100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813

PROGMAP

Use the PROGMAP command to display or place on the program stack information about programs currently loaded in storage.

Format

PROGMAP	<div style="display: flex; align-items: center;"> <div style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 10px; margin-right: 10px;"> <div style="border-bottom: 1px solid black; padding-bottom: 5px;"><i>programe</i></div> <div style="text-align: center; padding: 5px 0;">*</div> </div> <div style="margin-left: 10px;"> [(options...[])] </div> </div> <div style="margin-left: 20px;"> Options: </div> <div style="display: flex; align-items: center; margin-left: 40px;"> <div style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 10px; margin-right: 10px;"> PROGRAM NUCX ALL NOSEGment SEGment { <i>segname</i> } <div style="text-align: center; padding: 5px 0;">*</div> </div> <div style="margin-left: 10px;"> <div style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 10px; margin-right: 10px;"> STACK </div> <div style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 10px; margin-right: 10px;"> <div style="border-bottom: 1px solid black; padding-bottom: 5px;">FIFO</div> <div style="padding: 5px 0;">LIFO</div> </div> </div> </div>
----------------	---

Operands

programe

returns information about all programs with the specified program name.

- For a program loaded by the LOAD, INCLUDE, or LOADMOD commands, OS Simulation (LOAD or LINK ATTACH), or MODULEs invoked as commands (that have NOCLEAN attribute), *programe* is the file name.
- For a nucleus extension, *programe* is the command name.

An asterisk (*) will return information on all user programs, or nucleus extensions, or both. An asterisk is the default.

Options

PROGRAM

returns information for program(s) loaded by the LOAD, INCLUDE, or LOADMOD commands, OS Simulation (LOAD or LINK ATTACH), or MODULEs invoked as commands (that have NOCLEAN attribute). This is the default.

NUCX

returns information for nucleus extensions.

ALL

returns information for both programs and nucleus extensions.

NOSEGment

returns information on nucleus extensions that do not reside in loaded logical segments.

SEGment *segname*

returns information on nucleus extensions that reside in loaded logical segments. The *segname* specifies the 1-8 character name of a logical segment. An asterisk (*) indicates that all logical segments are to be searched for the nucleus extensions specified by *programe*.

STACK [FIFO]

STACK LIFO

specifies that CMS return information to the program stack rather than display the information at the terminal. The FIFO and LIFO options determine how the information is stacked. The default order is FIFO.

FIFO

stacks PROGMAP information first in first out on the program stack. The options STACK, STACK FIFO, and FIFO are equivalent.

LIFO

stacks PROGMAP information last in first out on the program stack. The options STACK LIFO and LIFO are equivalent.

Usage Notes

1. To display PROGMAP information at your terminal, omit the STACK, FIFO, and LIFO options. Use the STACK option to place the information on the program stack.
2. The ALL option displays information on programs and nucleus extensions. For a description of how to obtain information on NUCEXT look-aside entries, see the NUCXMAP command.

Examples

If you enter, PROGMAP PROG1, only information for PROG1 will be returned:

Name	Entry	Origin	Bytes	Attributes
PROG1	02000400	02000400	0000066D	AMODE 31 RELOC

Note: In these examples, the addresses above the 16Mb line would only be produced in a 370-XA environment.

If the command PROGMAP (ALL is entered, all programs and nucleus extensions are listed:

Name	Entry	Origin	Bytes	Attributes	
PROG1	02000400	04000400	0000066D	AMODE 31 RELOC	
PROG2	02000A6D	04000A6D	0000042A	AMODE 31 RELOC	
PROG3	00040000	00040000	00000338	AMODE 24 NON-RELOC	
Name	Entry	Userword	Origin	Bytes	Attributes
NUCX1	01000400	00004532	01000400	0000066D	AMODE ANY SYSTEM SERVICE
NUCX2	01000A6D	00000000	01000A6D	0000042A	AMODE 24
NUCX3	00020000	0000ABDC	00020000	00000338	AMODE 31 SYSTEM SERVICE

Note: All text decks are considered to be nonrelocatable programs whether or not you specified 'RLDSAVE' on the LOAD command. Therefore the 'NON-RELOC' attribute will be in the response from PROGMAP. However, when a module file is generated by GENMOD from a loaded text deck(s), the RLDSAVE/NORLDSAV option indicated during the LOAD process determines whether or not the module file will be relocated when it is loaded by LOADMOD.

Messages and Return Codes

- DMSPGM415E Invalid character *char* in program name *name* [RC = 20]
- DMSPGM941I User program *progid* is not loaded [RC = 0]
- DMSPGM942I No user programs are loaded [RC = 0]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

) |

Reason	Page
Errors in command syntax	811

PSERV

Use the PSERV command in CMS/DOS to copy, display, print, or punch a procedure from the VSE procedure library.

Format

PSERV	<i>procedure</i> [<i>ft</i> PROC] [(options... [])] <u>Options:</u> [DISK] [PRINT] [PUNCH] [TERM]
--------------	---

Operands*procedure*

specifies the name of the procedure in the VSE procedure library that you want to copy, print, punch, or display.

ft

specifies the file type of the file to be created on your disk or directory accessed as A. The *ft* defaults to PROC if a file type is not specified; the file name is always the same as the procedure name.

Options

You may enter as many options as you wish, depending on the functions you want to perform.

DISK

copies the procedure to a CMS file. If no options are specified, DISK is the default.

PRINT

spools a copy of the procedure to the virtual printer.

PUNCH

spools a copy of the procedure to the virtual punch.

TERM

displays the procedure on your terminal.

Usage Notes

1. You cannot execute VSE procedures in CMS/DOS. You can use the PSERV command to copy an existing VSE procedure onto a CMS disk or directory. Use the editor to change or add VSE job control statements to it, and then spool it to the reader of a VSE virtual machine for execution.
2. The PSERV command ignores current assignments of logical units, and directs output according to the option list.
3. The PSERV command does not support a private procedure library.

Responses

When you issue the TERM option, the procedure is displayed at your terminal.

Messages and Return Codes

DMSPRV003E	Invalid option: <i>option</i> [RC = 24]
DMSPRV004E	Procedure <i>procedure</i> not found [RC = 28]
DMSPRV006E	No read/write A filemode accessed [RC = 36]
DMSPRV070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSPRV097E	No SYSRES volume active [RC = 36]
DMSPRV098E	No procedure name specified [RC = 24]
DMSPRV099E	CMS/DOS environment not active [RC = 40]
DMSPRV105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSPRV113S	Disk(<i>vdev</i>) not attached [RC = 100]
DMSPRV411S	Input error code <i>nn</i> on SYSRES [RC = <i>rc</i>]

PUNCH

Use the PUNCH command to punch a CMS file to your virtual punch.

Format

PUnch	$fn\ ft\ \left[\begin{array}{c} fm \\ * \end{array} \right]\ \left[(\text{options... } []) \right]$
	<u>Options:</u> $\left[\begin{array}{c} \text{Header} \\ \text{NOHeader} \end{array} \right]\ \left[\text{MEMber } \left\{ \begin{array}{c} * \\ \text{membername} \end{array} \right\} \right]$

Operands

- fn**
is the file name of the file to be punched. This field must be specified.
- ft**
is the file type of the file to be punched. This field must be specified.
- fm**
is the file mode of the file to be punched. If you specify it as an asterisk (*), the standard order of search is followed and the first file found with the specified file name and file type is punched. If fm is not specified, your disk or directory accessed as A and its extensions are searched.

Options

Header

inserts a control card in front of the punched output. This control card indicates the file name and file type for a subsequent READCARD command to restore the file to a disk or directory. The control card format is shown in Table 15 on page 413.

NOHeader

does not punch a header control card.

MEMber *

MEMber membername

punches members of MACLIBs or TXTLIBs. If an asterisk (*) is entered, all individual members of that macro or text library are punched. If *membername* is specified, only that member is punched. If the file type is MACLIB and the **MEMBER membername** option is specified, the header contains **MEMBER** as the file type. If the file type is TXTLIB and the **MEMBER membername** option is specified, the header card contains **TEXT** as the file type.

Table 15. Header Card Format

Column	Number of Characters	Contents	Meaning
1	1	:	Identifies card as a control card.
2-5	4	READ	Identifies card as a READ control card.
6-7	2	blank	
8-15	8	<i>fname</i>	File name of the file punched.
16	1	blank	
17-24	8	<i>ftype</i>	File type of the file punched.
25	1	blank	
26-27	2	<i>fmode</i>	File mode of the file punched.
28	1	blank	
29-34	6	<i>volid</i>	Label of the disk from which the file was read or a “—” if read from a directory.
35	1	blank	
36-43	8	<i>mm/dd/yy</i>	The date that the file was last written.
44-45	2	blank	
46-50	5	<i>hh:mm</i>	The time of day that the file was written.
51-80	30	blank	

Usage Notes

1. You can punch fixed- or variable-length records with the PUNCH command, as long as no record exceeds 80 characters. Records with less than 80 characters are right-padded with blanks. Records longer than 80 characters are rejected. PUNCH changes variable-length record files to fixed-length 80-byte record files.
2. If you punch a MACLIB or TXTLIB file specifying the MEMBER * option, a READ control card is placed in front of each library member. If you punch a library without specifying the MEMBER * option, only one READ control card is placed at the front of the deck.
3. One spool punch file is produced for each PUNCH command; for example:

```
punch compute assemble (noh
```

punches the file COMPUTE ASSEMBLE, without inserting a header card. To transmit multiple CMS files as a single punch file, use the CP SPOOL command to spool the punch with the CONT option.

Note that if multiple files are sent with continuous spooling (using CP SPOOL PUNCH CONT) and a series of DISK DUMP commands, RECEIVE recognizes only the first file identifier (file name and file type). Any files having the same file identifier as existing files on your disk or directory accessed as A will overlay those files.

As a sender, you can avoid imposing this problem on file recipients by doing any of the following:

- a. Always use SENDFILE, which resets any continuous spooling options in effect.
- b. Do not spool the punch continuous.
- c. If you must send files with continuous spooling, warn the recipient(s) that files are being sent in this manner and list the file identifiers of the files you are sending.

Similarly, if the punch is spooled continuous and PUNCH is used to send multiple files, the file is read in as one file with “:READ” cards imbedded. In this case, although no files are overlaid, the recipient must divide the file into

individual files. This problem can also be avoided by using SENDFILE or by not spooling the punch continuous.

4. If the MEMBER option is specified more than once, only the last member specified will be punched. However, if one MEMBER option is coded with an asterisk (*), and another MEMBER option is specified with *membername*, only the member specified by *membername* will be punched, regardless of their order on the command line.

For example, if you code:

```
punch one maclib (member example1 member example2
```

only EXAMPLE2 will be punched. If you code:

```
punch one maclib (member example1 member *
```

only EXAMPLE1 will be punched.

5. When punching members from CMS MACLIBs, each member is followed by a // record, which is a MACLIB delimiter. You can edit the file to delete the // record.

Responses

None. The CMS ready message indicates that the command completed without error (the file was successfully spooled); the file is now under control of CP spooling functions. You may receive a message from CP indicating that the file is being spooled to a particular user's virtual reader.

Messages and Return Codes

DMSPUN002E	File <i>[fn [ft [fm]]]</i> not found [RC=28]
DMSPUN008E	Device <i>vdev</i> {invalid or nonexistent is an unsupported device type} [RC=36]
DMSPUN013E	Member <i>membername</i> not found [RC=32]
DMSPUN033E	File <i>fn ft fm</i> is not a library [RC=32]
DMSPUN039E	No entries in library <i>fn ft fm</i> [RC=32]
DMSPUN044E	Record exceeds allowable maximum [RC=32]
DMSPUN069E	Filemode <i>mode</i> not accessed [RC=36]
DMSPUN026E	Invalid parameter <i>parameter</i> in the option field <i>fieldname</i> [RC=24]
DMSPUN104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC=100]
DMSPUN123S	Error <i>nn</i> punching file <i>fn ft fm</i> [RC=100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

QUERY

Use the QUERY command to gather information about your CMS virtual machine. You can determine:

- The state of virtual machine characteristics that are controlled by the CMS SET command
- File definitions (set with the FILEDEF and DLBL commands) that are in effect
- The status of accessed disks and Shared File System (SFS) directories
- The search order for libraries (MACLIBs, TXTLIBs, CSLLIBs, DOSLIBs, and LOADLIBs)
- The status of CMS/DOS functions
- Saved segment information
- Authorities granted for SFS directories or files in SFS directories
- Aliases to base files in directories
- Information about your SFS file pool
- Locks on SFS directories or files in SFS directories

Other CMS QUERY operands give you the following information about virtual screens and windows in a full-screen CMS environment:

- Physical screen characteristics
- CMSPF and WMPF key settings
- Window characteristics and the order in which the windows are being displayed
- Virtual screen characteristics
- Cursor position

See “QUERY” on page 704 for the virtual screen and windowing CMS QUERY operands.

QUERY

Query	<p> ABBREV ACCESSED $\left[\begin{array}{c} fm \\ * \\ \overline{R/O} \\ R/W \end{array} \right]$ </p> <p> ALIAS $\left\{ \begin{array}{c} fn \ ft \\ * \ * \end{array} \right\} [dirid]$ </p> <p> APL AUThority $\left[\begin{array}{c} fn \ ft \\ * \ * \end{array} \right] [dirid]$ </p> <p> AUTOREAD BLIP CMSLEVEL CMSTYPE COMDIR CSLLIB DISK $\left[\begin{array}{c} fm \\ * \\ \overline{R/W} \\ MAX \\ FIRSTR/W \end{array} \right]$ </p> <p> DLBL DOS DOSLIB DOSLNCNT DOSPART ENROLL $\left\{ \begin{array}{c} USEr \\ ADMinistrator \end{array} \right\} \text{ FOR } \left\{ \begin{array}{c} userid \\ nickname \\ ALL \end{array} \right\} [filepoolid:]$ </p> <p> EXECTRAC FILEDEF FILEPOOL $\left\{ \begin{array}{c} CURrent \\ PRImary \\ CONFLict \left\{ \begin{array}{c} userid \\ nickname \end{array} \right\} [filepoolid:] \\ \\ CONNect \text{ FOR } \left\{ \begin{array}{c} userid \\ nickname \\ ALL \end{array} \right\} [filepoolid:] \end{array} \right\}$ </p> <p> FILEWAIT FULLREAD IMESCAPE IMPCP IMPEX INPUT INSTSEG KEYPROTect </p>	<p>[(options ...[)]]</p>
-------	---	----------------------------

<p>Query</p>	<pre> LABELDEF LANGLIST LANGUAGE [ALL] LDRTBLS LIBRARY LIMITS * [filepoolid:] LINEND LOADAREA LOADLIB LOCK [fn ft] [dirid] [* *] MACLIB NAMEDEF NONDISP OPTION OUTPUT PROTECT RDYMSG REDTYPE RELPAGE REMOTE SEARCH SEGMENT [segname [CONTENTs Assign SPACE PHysical LOgical]] [* [SPACE PHysical LOgical]] SERVER STORECLR SYNONYM { SYSTEM USER ALL } SYSNAMES TEXT TRANslate [SYStem [TRANslate [APPLID applid]]] [USER [SYNonym [*]]] [ALL [BOTH]] TXTLIB UPSI </pre> <p style="text-align: right;">} [(options ... [])]</p> <p style="text-align: center;">Options: [STACK [FIFO LIFO]] [FIFO LIFO XEDIT ¹]</p> <p>¹ XEDIT option is only for QUERY ALIAS and QUERY AUTHORITY.</p>
---------------------	---

ABBREV

displays the status of the minimum truncation indicator.

Response:

ABBREV = ON

or

ABBREV = OFF

where:

ON

indicates that truncations are accepted for CMS commands and all translations.

OFF

indicates that truncations are not accepted.

ACCESSED *fm*

displays status of a single disk or Shared File System (SFS) directory represented by *fm*.

Response:

Mode	Stat	Files	Vdev	Label/Directory
mode	stat	files	vdev	label/directory

Note: The header is generated only if output is displayed at the terminal.

where:

mode

is the file mode letter.

stat

is the status of the disk or SFS directory: R/O (read-only) or R/W (read/write).

files

is the number of files on the disk or the number of base files, aliases, subdirectories, erased aliases, and revoked aliases in the directory. This number is the same as the number of files that would be displayed if you issued LISTFILE * * *fm* (ALLFILE).

vdev

is either the virtual address of the device if the entry is a disk or is 'DIR' if the entry is a directory.

label/directory

is either the label assigned to the CMS disk when it was formatted or the complete SFS directory name.

If the entry is an OS or DOS disk, this is the volume label.

Note: If you are in full-screen CMS and a directory name is too long to be displayed in the window, you can scroll to the right to see the remainder of the directory name.

If the disk or directory with the specified file mode is not accessed, the response is:

Disk *fm* not found

ACCESSED *

displays status of all your accessed CMS disks or SFS directories. This is the default.

Response: The response is the same as for QUERY ACCESSED *fm*; one line is displayed for each accessed disk or directory.

ACCESSED R/O

displays status of your accessed read only CMS disks or SFS directories.

Response: The response is the same as for QUERY ACCESSED *fm*; one line is displayed for accessed disk or directory that is read only.

If there are no disks or directories accessed in read only mode, the response is:

No filemode is read/only

ACCESSED R/W

displays status of your accessed read/write CMS disks or SFS directories.

Response: The response is the same as for QUERY ACCESSED *fm*; one line is displayed for each accessed disk or directory that is read/write.

If there are no disks or directories accessed in read/write mode, the response is:

No read/write disk accessed

ALIAS *fn ft [dirid]*

displays alias information for a base file or an alias in a Shared File System (SFS) directory. If *dirid* is not specified, the directory queried is the one accessed as A. See "Naming Shared File System (SFS) Directories" on page 4 for details on how to specify *dirid*. You must have read or write authority to the file you are querying, and the specified directory cannot be open.

For base files:

- If you are the owner, information is returned listing the users that have aliases to the base file along with the number of aliases they have to the base file. If you own or have read or write authority on the directory containing the alias, then the alias name is also returned.
- If you are not the owner, information is returned listing your aliases to the base file.

For aliases:

- Information is returned listing the owner of the base file. If you own or have read or write authority on the directory containing the base file then the base file name is also returned.
- For a revoked or erased alias, the owner of the base file is returned. For a revoked alias, if you own or have read or write authority on the directory containing the base file then the base file name is also returned.

You can use special characters (* or %) to query a set of files. If a special character is specified, then all aliases and base files that have aliases to them that match the specified criteria are listed. Only those files for which you have read or write authority are listed. See "Pattern Matching" on page 8 for more detail on pattern matching.

Note: From the XEDIT environment, the XEDIT option is valid for QUERY ALIAS. Use the XEDIT option to place the output in a file that you are currently displaying. Your file record format must be variable or fixed with a record length of 190. See "Options" on page 451 for more detail.

Response:

```
Directory = filepoolid:userid.n1.n2...n8
Filename Filetype Fm T Userid Num Filename Filetype Directory
fn1      ft1      fm t userid  nnn fn2      ft2      dirname
fn1      ft1      fm t userid  nnn fn2      ft2      dirname
.        .        . . .      . .        .        .
.        .        . . .      . .        .        .
.        .        . . .      . .        .        .
```

Note: The second line of the header is generated only if output is displayed at the terminal; the first line containing the directory name is always included.

where:

fn1
is the file name of the file queried.

ft1
is the file type of the file queried.

fm
is the file mode of the directory if it is accessed, otherwise, this column contains a dash (-).

t
is the type of file for **fn1 ft1** as follows:

- A = alias
- B = base file
- E = erased alias (the base file has been erased)
- R = revoked alias (authority to base file has been revoked)

userid
is either the user that has an alias to the file if you are querying a base file, or, if you are querying an alias, then **userid** is the owner of the base file.

nnn
is the number of aliases to the base file that the user has created. This number is 1 if you own the directory or have read or write authority to the directory that contains the alias, and the name of the alias is shown in the *fn2 ft2* columns.

fn2
is the file name of an alias to the base file if you are querying a base file. If you are querying an alias, then this is the file name of the base file.

ft2
is the file type of an alias to the base file if you are querying a base file. If you are querying an alias, then this is the file type of the base file.

dirname
is the name of the directory containing **fn2 ft2**.

If no aliases are in effect, and the **STACK**, **LIFO**, or **FIFO** option was specified, the return code is set to 6, indicating that no data was stacked.

Example:

Smith queries for aliases on the base file **CODING STANDRDS** in the **.INFO** directory:

```
query alias coding standrds smith.info
```

and gets the following response:

```
Directory = FILEPOOL:SMITH.INFO
Filename Filetype Fm T Userid Num Filename Filetype Directory
CODING STANDRDS A0 B SMITH 1 CODE REQMTS .STANDARDS.V1
CODING STANDRDS A0 B DAVIS 2
CODING STANDRDS A0 B HAYES 1 BAL STANDARDS .
```

This response tells SMITH that there are four aliases for his base file CODING STANDRDS. The first three columns identify the queried file, column T has a 'B' indicating that CODING STANDRDS is a base file. Column NUM shows the number of aliases each user has to the base file and the last three columns indicate the name of the aliases. These last three columns are blank if SMITH does not have authority to the directory that contains the aliases.

In the next example, SMITH queries the alias MACRO in his .INFO directory, query alias macro * smith.info

and gets the following response,

```
Directory = FILEPOOL:SMITH.INFO
Filename Filetype Fm T Userid Num Filename Filetype Directory
MACRO STANDRDS A0 A JOHNN 1 MACRO INFORMTN .
MACRO OLDSTNRD A0 E JOHNN 1
MACRO PROPOSED A0 R JOHNN 1
```

SMITH has three aliases that match this query. The first is an alias to MACRO INFORMTN in the user JOHNN's top directory. For the alias MACRO OLDSTNRD, the 'E' in column T indicates that the base file has been erased. For MACRO PROPOSED, an 'R' in column T indicates that SMITH's authority to the base file has been revoked. If the last three columns are blank, this indicates that SMITH does not have authority to the directory containing the base file.

APL

displays the status of APL character code conversion.

Response:

APL ON

or

APL OFF

where:

ON

converts APL characters for windows.

OFF

does not convert APL characters.

AUTHORITY [*fn ft*] [*dirid*]

displays authorities for a Shared File System (SFS) directory or for a file(s) in the directory. If you own the file or directory, a list of user IDs, including your own, is returned with the authority that was granted to the file or directory. If you are not the owner of the file or directory, only your authority on the file or directory is returned.

If *fn ft* is not specified, the authority you have on the directory is shown. If *fn ft* is specified as asterisks, all the files contained in the *dirid* for which you have read or write authority are shown. If *dirid* is omitted, the directory queried is the one accessed as A.

QUERY

You must have read or write authority for the specified file and the directory cannot be open. If the file is an alias, the alias is displayed with the authority that you have on the base file.

Special characters (* or %) can be used to designate a set of files, providing you have read or write authority for the directory specified by *dirid*. For further description of *dirid* see "Naming Shared File System (SFS) Directories" on page 4. For more on using special characters to do pattern matching, see "Pattern Matching" on page 8.

When pattern matching is done on subdirectory names that contain more than eight characters, the first eight characters are used as the file name and the remaining characters are used as the file type. For example, your directory accessed as A contains,

```
CROCKETT NOTEBOOK
CROCKETTNOTES
```

where CROCKETT NOTEBOOK is a file and CROCKETTNOTES is a subdirectory. Issuing,

```
query authorit croc* n* a
```

would find both CROCKETT NOTEBOOK and CROCKETTNOTES because CROCKETTNOTES contains more than eight characters and is matched as if it had a file name of CROCKETT and a file type of NOTES. Issuing,

```
query authorit crockettnotes * a
```

would also find both CROCKETT NOTEBOOK and CROCKETTNOTES because only the first eight characters are used as the file name. Therefore, any file or alias with the name of CROCKETT, or any subdirectory with CROCKETT as the first eight characters in its name would be listed.

Note: From the XEDIT environment, the XEDIT option is valid for QUERY AUTHORITY. Use the XEDIT option to place the output in a file that you are currently displaying. See "Options" on page 451 for more detail.

Response:

```
For QUERY AUTHORITY fn ft [dirid]
```

```
Directory = filepool:userid.n1.n2...n8
Filename Filetype Fm Type   Grantee R W
fn      ft      fm type  userid  r w
fn      ft      fm type  userid  r w
.       .       . .      .       . .
.       .       . .      .       . .
.       .       . .      .       . .
```

Note: The second line of the header is generated only if output is displayed at the terminal; the first line containing the directory name is always included. For QUERY AUTHORITY *dirid*

```
Directory = filepool:userid.n1.n2...n8
Grantee  R  W
userid   r  w
.        .  .
.        .  .
.        .  .
```

where:

fn
is the name of the file or subdirectory. If you have queried a directory, this field is omitted.

ft
is the file type. If you have queried a directory, this field is omitted.

fm
is the file mode of the directory where the file is contained. If the directory is not accessed, then this column contains a dash. If you have queried a directory, this field is omitted.

type
is the type of file or directory as follows:

ALIAS = alias for a base file

BASE = base file

DIR = directory

ERASED = erased (the base file has been erased)

REVOKED = revoked (authority to the base file has been revoked).

userid
is the user ID that has been granted authority on the file or directory. This will be **<PUBLIC>** if authority was granted to all users who can connect to the file pool.

r
indicates the read authority for the listed file or directory.

X
means you have the authority.

—
means you do not have the authority or that the authority is revoked.

P
means that the authority is held by the base security manager and the listed authority may not be accurate. The External Security Manager (ESM) must be queried to get the actual authorities.

w
indicates the write authority for the listed file or directory.

X
means you have the authority.

—
means you do not have the authority or that the authority is revoked.

P
means that the authority is held by the base security manager and the listed authority may not be accurate. The External Security Manager (ESM) must be queried to get the actual authorities.

Example:

John wants to see the authorities he has been granted on another user's directory, so he issues,

```
query authorit * * smith.info
```

QUERY

and the following information is displayed,

```
Directory = FILEPOOL:SMITH.INFO
Filename Filetype Fm Type Grantee R W
EXAMPLE SCRIPT A1 BASE JOHN X X
PROJ2 DIR JOHN X -
CODING STANDRDS A0 ALIAS JOHN XP -P
MACRO PROPOSED A0 REVOKED JOHN - -
```

John has read/write authority to the file EXAMPLE SCRIPT. He has read authority to the directory SMITH.INFO.PROJ2. There is an alias, CODING STANDRDS, that he was granted read authority on, but the authority is ESM-protected. John's authority to use the alias, MACRO PROPOSED, has been revoked.

AUTOREAD

displays the status of the console read.

Response:

```
AUTOREAD = ON
```

or

```
AUTOREAD = OFF
```

where:

ON indicates that a console read is issued immediately after command execution.

OFF indicates that no console read is issued until the enter key (or its equivalent) is pressed.

BLIP

CMS does not support the BLIP facility. The command is maintained for compatibility only.

Response:

```
BLIP = OFF
```

CMSLEVEL

returns the feature or program product, release, and the service level of CMS.

Response: Displays the VM/SP Release Level and the Service Level.

such as:

```
VM/SP RELEASE 5, SERVICE LEVEL 103
```

CMSTYPE

indicates the status of the CMS terminal display. This option is valid only from an exec environment. The STACK and/or LIFO or FIFO options must be specified.

Response: CMSTYPE = {HT|RT}

where:

HT

indicates that the CMS terminal display within an exec is suppressed. All CMS terminal display from an exec, except for CMS error messages with a suffix letter of 'S' or 'T', is suppressed until the end of the exec file or until the SET CMSTYPE RT command is executed.

RT

indicates that the CMS terminal display is not suppressed.

COMDIR

displays the current CMS communications directory settings. The settings can be changed using the SET COMDIR command.

Response:

```
SYSTEM status STATIC fileid
USER status STATIC fileid
```

where:

SYSTEM

indicates the system, or secondary, communications directory.

USER

indicates the user, or primary, communications directory.

status

indicates the status of the name resolution. The *status* is either:

ON

indicating that name resolution is on.

OFF

indicating that name resolution is off.

STATIC

indicates that the file was loaded into memory when the last SET COMDIR FILE or SET COMDIR RELOAD was entered. If you have changed either file since then, enter the SET COMDIR RELOAD command to apply the changes.

fileid

is the name of the directory. A response of "(none)" indicates that no directory is loaded.

CSLLIB

displays the names of the callable services libraries, with a file type of CSLLIB or CSLSEG, in the current library search order. (This means all callable services libraries specified on the last GLOBAL CSLLIB command, if any.)

Response:

```
CSLLIB = libname1 . . . libname8
. . . . .
. . . . .
. . . . .
```

Up to eight names are displayed per line, for as many lines as necessary. If no CSLLIBs are found in the search order, the following message is displayed at the terminal:

```
CSLLIB = NONE
```

DISK *fm*

displays the status of a disk or Shared File System (SFS) directory represented by *fm*, file mode letter.

Response:

QUERY

```
| LABEL VDEV M STAT CYL TYPE BLKSIZE FILES BLKS USED-(%) BLKS LEFT BLK TOTAL  
| label vdev m stat cyl type blksize files blks_used blks_left blk_total
```

If the disk is an OS or DOS disk, the response is:

```
| LABEL VDEV M STAT CYL TYPE BLKSIZE FILES BLKS USED-(%) BLKS LEFT BLK TOTAL  
| label vdev m stat cyl type files
```

where:

label

is either the label assigned to the disk when it was formatted or the volume label if it is an OS or DOS disk. If it is an SFS directory, a dash is displayed in the label column.

vdev

is either the virtual device address for a disk or DIR for an SFS directory.

m

is the file mode letter.

stat

indicates whether a disk or SFS directory status is read/write (R/W) or read/only (R/O).

cyl

is the number of cylinders available on the disk. For an FB-512 device, this field contains the notation FB rather than the number of cylinders. For an OS-formatted disk the number of cylinders available may appear to be larger than the number actually available, because it includes the space used by the VTOC. A dash is displayed for an SFS directory.

type

is the device type of the disk or a dash is displayed for an SFS directory.

blksize

is the CMS disk block size when the minidisk was formatted. For an SFS directory, the block size is always 4096.

files

is the number of CMS files on the disk or the number of base files, aliases, subdirectories, erased aliases, and revoked aliases in the directory. For an OS or DOS disk, OS or DOS is displayed.

blks_used

indicates the number of CMS disk blocks in use. The CMS disk blocks include both data blocks and control blocks. The percentage of blocks in use is also displayed. A dash is displayed for an SFS directory.

blks_left

indicates the number of disk blocks left. This is a high approximation because control blocks are included. A dash is displayed for an SFS directory.

blk_total

indicates the total number of disk blocks. A dash is displayed for an SFS directory.

If the disk or directory with the specified file mode is not accessed, the response is:

Disk *fm* not accessed

DISK *

displays the status of all accessed CMS disks and SFS directories. This is the default.

Response: Header is the same as QUERY DISK *fm* and QUERY DISK R/W. One line is displayed for each accessed disk or directory.

DISK R/W

displays the status of all CMS disks and SFS directories that have been accessed in read/write mode.

Response: Header is the same as QUERY DISK *fm* and QUERY DISK *. One line is displayed for each read/write disk and directory.

If there are no disks or directories accessed in read/write mode, the response is:

No read/write disk accessed

DISK MAX

displays the status of a CMS disk or SFS directory accessed in read/write mode having the most available space. All directories in the same file space share the space, therefore the available space for a directory is based on the amount of space available in the file space. If communication fails between this command and the file pool, CMS assumes there is at least one block of read/write storage for a directory that is accessed as read/write.

Response: Is the same format as QUERY DISK *fm*; a header and one line is displayed for the read/write disk and directory with the most available space.

If there are no disks or directories accessed in read/write mode, the response is:

No read/write disk accessed

If there is no space available on any of the disks or directories accessed in read/write mode, the response is:

No read/write disk or directory with space is accessed

DISK FIRSTR/W

displays the status of the first CMS disk or SFS directory in the search order accessed in read/write mode.

Response: Is the same format as QUERY DISK *fm*; a header and one line are displayed for the first read/write disk or directory.

If there are no disks or directories accessed in read/write mode, the response is:

No read/write disk accessed

DLBL

in order to display the contents of the current data set definitions, it is necessary only to enter:

DLBL or QUERY DLBL

Entering the command yields the following information:

DDNAME

the VSE file name or OS ddname.

QUERY

MODE

the CMS file mode identifying the disk or directory on which the data set resides.

LOGUNIT

the VSE logical unit specification (SYSxxx). This operand will be blank for a data set defined while in CMS/OS environment; that is, the SET DOS ON command had not been issued at DLBL definition time.

TYPE

indicates the type of data set defined. This field may only have the values SEQ (sequential) and VSAM.

CATALOG

indicates the ddname of the VSAM catalog to be searched for the specified data set. This field will be blank for sequential (SEQ) dataset definitions.

EXT

specifies the number of extents defined for the data set. The actual extents may be displayed by entering either the DLBL (EXTENT) or the QUERY DLBL EXTENT command. This field will be blank if no extents are active for a VSAM data set or if the data set is sequential (SEQ). If no DLBL MULT definitions are active, the response is:

No user defined MULTs in effect.

VOL

specifies the number (if greater than one) of volumes on which the VSAM data set resides. The actual volumes may be displayed by entering either the DLBL (MULT) or the QUERY DLBL MULT commands. This field will be blank if the VSAM data set resides only on one volume or if the data set is sequential (SEQ). If no DLBL EXTENT definitions are active, the response is:

No user defined EXTENTS in effect.

BUFSP

indicates the size of the VSAM buffer space if entered at DLBL definition time. This field will be blank if the dataset is sequential (SEQ).

PERM

indicates whether the DLBL definition was made with the PERM option. The field will contain YES or NO.

DISK

indicates whether the data set resided on a CMS disk (or directory) or a DOS/OS disk at DLBL definition time. The values for this field are DOS and CMS.

DATASET.NAME

for a data set residing on a CMS disk or directory, the CMS file name and file type are given; for a data set residing on a DOS/OS disk, the data set name (maximum 44 characters) is given. This field will be blank if no DOS/OS data set name is entered at DLBL definition time. If no DLBL definitions are active, the following message is issued:

No user defined DLBL in effect.

DOS

displays whether the CMS/DOS environment is active or not.

Response:

DOS = ON

or

DOS = OFF

DOSLIB

displays the names of all files with a file type of DOSLIB that are to be searched for executable phases (that is, all DOSLIBs specified on the last GLOBAL DOSLIB command, if any).

Response:

DOSLIB = libname1 ... libname8

```

.      .      .
.      .      .
.      .      .
    
```

Up to eight names are displayed per line, for as many lines as are necessary. If no DOSLIBs are to be searched, the response is:

DOSLIB = NONE

DOSLNCNT

displays the number of SYSLST lines per page.

Response: DOSLNCNT = nn

where:

nn is an integer from 30 to 99.

DOSPART

displays the current setting of the virtual partition size.

Response:

nnnnnK

or

NONE

where:

nnnnnK

indicates the size of the virtual partition to be used at program execution time.

NONE

indicates that CMS determines the virtual partition size at program execution time.

ENROLL USER FOR ALL [filepoolid:]

displays information about enrolled users in a specified file pool. If *filepoolid:* is not specified, the default file pool is used.

Note: If the QUERY ENROLL command is issued from an exec on a work unit that is active, the command will fail.

Response:

Number Of Enrolled Users = nnnn

userid

```

.
.
.
    
```


QUERY

Note: If the file pool administrator has issued a public enrollment for the file pool, <PUBLIC> is displayed as one of the user IDs. If there are no users enrolled, you will get the following response:

No users are enrolled in file pool *filepoolid*

with a return code of 0 if the response is typed or a return code of 6 if the response was to be stacked.

ENROLL USER FOR *userid/nickname [filepoolid:]*

displays whether the specified user or group of users is enrolled. Nicknames can be used, see the NAMES command. If *filepoolid:* is not specified, the default file pool is used.

Response:

Userid	Enrolled
userid	YES/NO
.	.
.	.
.	.

Note: The header is generated only if output is displayed at the terminal.

ENROLL ADMINISTRATOR FOR ALL [*filepoolid:*]

displays user IDs of all enrolled file pool administrators. If *filepoolid:* is not specified, the default file pool is used.

Response:

Number Of Administrators = nn
userid
.
.
.

If there are no administrators enrolled, you will get the following response:

There are no administrators for file pool *filepoolid*

with a return code of 0 if the response is typed or a return code of 6 if the response was to be stacked.

ENROLL ADMINISTRATOR FOR *userid/nickname [filepoolid:]*

displays whether the specified user(s) are file pool administrators or not. If *filepoolid:* is not specified, then the default file pool is used. Nicknames can be used, see the NAMES command.

Response:

Userid	Administrator
userid	YES/NO
.	.
.	.
.	.

Note: The header is generated only if output is displayed at the terminal.

EXECTRAC

displays the setting of the tracing bit (in EXECFLAG in NUCON). Setting is either ON or OFF.

Response:

EXECTRAC = ON

or

EXECTRAC = OFF

FILEDEF

displays all file definitions in effect.

Response:

```
ddname device
.
.
.
```

If the device is a disk, additional file ID information is displayed:

```
ddname DISK filename filetype filemode [datasetname]
```

or, if the device is a tape, additional label information is displayed:

```
ddname TAPn labeltype [n [VOLID valid]|filename]
```

If no file definitions are in effect, the following message is displayed at the terminal:

No user defined FILEDEFs in effect

FILEPOOL CURrent

displays the current default file pool set by the SET FILEPOOL command.

Response:

```
FILEPOOL filepoolid:
```

If there is no current file pool, the response is:

```
FILEPOOL NONE
```

FILEPOOL PRImary

displays the primary file pool specified during IPL.

Response: The response is the same as for QUERY FILEPOOL CURRENT.

FILEPOOL CONFLict *userid***]***nickname* [*filepoolid:*]

displays information about lock conflicts in the specified file pool for the specified user ID or set of users. To specify a nickname for a set of users, see the NAMES command. If you do not specify *filepoolid:*, then the default file pool is queried.

You cannot use this query to find your own conflicts. A response indicates that you have no conflicts. When you are in a lock wait state, i.e., a filepool conflict, your virtual machine waits for the file pool server to reply and you cannot enter any commands until the conflict is resolved.

Response:

Requestor	Holder	Wait	Lock	Lock Type
requestor	holder	wait	lock	lock type
.
.
.

Note: The header is generated only if output is displayed at the terminal.

where:

requestor

is the user who is waiting for a lock that is being held by the holder.

holder

is the user who is preventing the requestor from getting a lock. The holder may be waiting for some other user to free a lock.

wait

is one of the following wait states of the holder.

Communication means that the file pool server is waiting for another request from the holder of the lock.

Checkpoint means that the holder of the lock is waiting for an internal checkpoint to complete.

Lock means that the holder of the lock is waiting for another implicit lock that someone else holds. The file pool server never waits for explicit locks, which are created by the **CREATE LOCK** command.

Catalog Buffer means that the holder of the lock is waiting for a buffer to hold catalog minidisk data.

Control Buffer means that the holder of the lock is waiting for a buffer to hold control minidisk data.

I/O means that the holder of the lock is waiting for file pool I/O to complete.

None there is no wait state.

lock

one of the following types of file pool resource for which the Requestor has requested a lock.

- File
- Catalog Row
- Catalog Block
- Catalog Index
- Storage Group
- Recovery
- File Space
- Directory

lock type

is the type of lock that the requestor needs. Some of these are internal lock types that the server needs to complete a request. The internal lock types are intended for use by service personnel. "Lock Type" can be:

Share is an internal locking type.

Excl is an internal locking type.

IShare is an internal locking type.

IExcl is an internal locking type.

Ck_Shr is an explicit (checkout) share lock.

Ck_Up
is an explicit (checkout) update lock.

Ck_Ex
is an explicit (checkout) exclusive lock.

Sh_Ix
is an internal locking type.

FILEPOOL CONNect FOR ALL [filepoolid:]

displays information about all the users that are currently connected to the specified file pool. If *filepoolid:* is not specified, the default file pool is queried.

Response:

Userid	Connected
userid	connected
.	.
.	.
.	.

Note: The header is generated only if output is displayed at the terminal.

where:

userid
is the user ID of the user connected to the file pool.

Note: A single user ID may have multiple connections and would be listed for each connection.

connected
is one of the following:

Yes = connected to file pool and does not have an active request.

No = not connected (not applicable if ALL is specified)

Active = connected to file pool and has an active request

FILEPOOL CONNect FOR *userid*/*nickname* [filepoolid:]

displays information about a user's connection to a specified file pool. If *filepoolid:* is not specified, the default file pool is queried. The *userid* can be a nickname (see the NAMES command).

Response: The same response format as for QUERY FILEPOOL CONNECT FOR ALL; the Connected column can also contain a No indicating that the user ID is not connected to the file pool.

FILEWAIT

displays whether or not you want commands affecting files in a filepool to wait for a locked file to become available. The setting can be changed using the SET FILEWAIT command.

Response:

FILEWAIT ON|OFF

where

ON
indicates that you do not want commands affecting a file to fail because the file is unavailable. The request waits until the file becomes available. The wait may be for a prolonged time.

OFF

means that commands affecting a file will fail immediately if the requested file is not available.

FULLREAD

indicates whether or not 3270 null characters are recognized in the middle of the physical screen.

Response:

FULLREAD ON

or

FULLREAD OFF

where:

ON

indicates that null characters are recognized in the middle of lines, making it easier for you to enter tabular or pictorial data.

OFF

inhibits transmission of nulls from the terminal.

IMESCAPE

displays the immediate command escape character.

Response:

IMESCAPE = char

or

IMESCAPE = OFF

where:

char

is the escape character in effect. The default character is a semi-colon (;).

OFF

no immediate command escape character is in effect.

IMPCP

displays the status of implied CP command indicator.

Response:

IMPCP = ON

or

IMPCP = OFF

where:

ON

indicates that CP commands can be entered from the CMS environment.

OFF

indicates that you must use the CP command or the CP commands from the CMS environment.

IMPEX

displays status of implied exec indicator.

Response:

IMPEX = ON

or

IMPEX = OFF

where:

ON

indicates that exec files can be executed by entering the filename of the file.

OFF

indicates that the EXEC command must be explicitly entered to execute exec files.

INPUT

displays the contents of any input translate table in effect.

Response:

```
INPUT a1 xx1
      . .
      . .
      . .
      an xxn
```

If you do not have an input translate table in effect, the response is:

NO USER DEFINED INPUT TRANSLATE TABLE IN USE

INSTSEG

displays the status of the CMS installation saved segment.

Response:

INSTSEG = ON mode|LAST

or

INSTSEG = OFF

where:

ON mode|LAST

indicates that the installation saved segment is being used. The *mode* specifies the location in the command search order where the saved segment is being accessed. LAST indicates that the CMS installation saved segment is searched after the last accessed disk or directory in the search order.

OFF

indicates that the CMS installation saved segment is not being used.

KEYPROTECT

checks the setting of the storage keys according to their defined values.

Response:

```
KEYPROT = ON
          OFF
```

where:

ON

indicates that the storage keys were reset.

OFF

indicates that the storage keys were not reset.

LABELDEF

displays all label definitions in effect.

Response:

DDNAME	VOLID	FSEQ	VOLSEQ	GENN	GENV	CRDTE	EXDTE	SEC	FID
ddname	volid	fseq	volseq	genn	genv	crdte	exdte	sec	fid
.
.
.

Only fields you have explicitly specified are displayed. Defaulted fields are not displayed. If no label definitions are in effect, the following message is displayed at the terminal:

No user defined LABELDEFs in effect

For an OS simulation user, if SCRATCH was entered at command time and the file has not been opened, then all the volume IDs and SCRATCH will be displayed. If SCRATCH was entered at command time, and the file has been opened, then all the volume IDs and the volume IDs of all the scratch tapes will be displayed. When multiple volume IDs have been specified, the response is:

DDNAME	VOLID	FSEQ	VOLSEQ	GENN	GENV	CRDTE	EXDTE	SEC	FID
ddname		fseq	volseq	genn	genv	crdte	exdte	sec	fid
VOLIDS:									
.
.

LANGLIST

displays all the language identifiers (*langids*) that can be set for CMS in your virtual machine. You can use this command to determine whether or not a certain language is valid for your virtual machine.

Response:

langid1
langid2
langid3

⋮

where:

langid1

is the language identifier of the default language for CMS.

langid2

is the language identifier of the language that is currently active for CMS in your virtual machine. If the current, active language and the default language are the same, *langid2* is not displayed.

langid3...

are other valid language identifiers.

Note: The response may change if you alter the size of your virtual machine.

Example:

The response to QUERY LANGLIST may be:

AMENG
GER
FRANC

where:

AMENG is American English, the default language for CMS.
GER is German, the language that is currently active for CMS in your virtual machine.
FRANC is French, another valid language for your virtual machine.

LANGUAGE

displays the language identifier of the language that is currently active for CMS in your virtual machine.

Response: langid

where:

langid

is the language identifier of the language that is currently active for CMS in your virtual machine.

LANGUAGE ALL

displays the language identifier and all active application identifiers. For applications, **ALL** displays whether the applications have system-provided language files, user additions (message repositories, CMS command syntax file), or both.

Response:

langid
applid1
 USER|SYSTEM|ALL
applid2
 USER|SYSTEM|ALL
 ⋮

where:

langid

is the language identifier of the language that is currently active for CMS in your virtual machine.

applid1 applid2 ...

is an application identifier.

USER

indicates that user repositories, command syntax tables, and/or command synonym tables are loaded into storage.

SYSTEM

indicates that the system-provided language files for the application named are active. No user language files are active.

ALL

specifies that the system-provided language files *and* user additions are active.

Example:

The response to **QUERY LANGUAGE ALL** may be:

QUERY

AMENG
DMS
ALL
AWG
SYSTEM

where:

AMENG is the language ID for American English.
DMS is the application id for CMS.
ALL indicates that you have made additions to the CMS message repository and/or command syntax files.
AWG is an application id for an application program.
SYSTEM indicates that you are using only the system language information for the AWG application.

LDRTBLS

displays the number of loader tables.

Response:

LDRTBLS = nn

LIBRARY

displays the names of all library files with file types of MACLIB, TXTLIB, CSLLIB, DOSLIB, and LOADLIB that are to be searched.

Response:

```
libtype = libname1 ... libname8  
          .           .  
          .           .  
          .           .
```

Up to eight names are displayed per line, for as many lines as necessary. If no libraries are to be searched, the response is:

```
MACLIB = NONE  
TXTLIB = NONE  
DOSLIB = NONE  
LOADLIB = NONE  
CSLLIB = NONE
```

LIMITS * [*filepoolid:*]

displays information about your limits in the specified file pool. If *filepoolid:* is not specified, the default file pool is used.

If you are a remote or batch user, the user ID that is queried is your APPC/VM access user ID. This is the user ID by which the file pool knows you and is used to verify authorization.

Note: If the QUERY LIMITS command is issued from an exec on a work unit that is active, the command will fail.

Response:

Userid	Storage Group	4K Block Limit	4K Blocks Committed	Threshold
userid	stor_group	blk_limit	blks_com	thresh

Note: The header is generated only if output is displayed at the terminal.

where:

userid
is your user ID.

stor_group

is the number of the Storage Group to which you have been assigned.

blk_limit

is the number of 4K blocks available for your files. This number is the size of your file space.

blks_com

is the number of 4K blocks committed in your file space and the percentage of file space used.

thresh

is the percentage of the **blk_limit** at which you will receive a warning message if you use this percentage or more of your file space. The default value is 90%. This value can be changed by the **SET THRESHOLD** command.

If you are not enrolled in the file pool, the response is your user ID and dashes in all the other columns.

LINEND

indicates whether or not the logical line end character is activated and the character defined as the logical line end.

Response:

LINEND ON char

or

LINEND OFF char

where:

ON

indicates that the logical line end character is activated.

OFF

indicates that the logical line end character is not activated.

char

is the logical line end character.

LOADAREA

displays the default ORIGIN for loading TEXT files with the LOAD command.

Note: Use the SET LOADAREA command to change the default ORIGIN.

Responses:

LOADAREA = 20000

or

LOADAREA = RESPECT

where:

20000

indicates that the LOAD command will start loading at X'20000' if the ORIGIN or RMODE option is not specified on the LOAD command.

RESPECT

indicates that in a 370-XA mode virtual machine the LOAD command will respect the RMODE during TEXT ESD processing to determine where the loaded program should reside when the RMODE or ORIGIN option is not specified on the LOAD command.

In a VM/SP or System/370 mode virtual machine, the SET LOADAREA RESPECT command will cause loading to start at the beginning of the largest contiguous area below 16Mb if the ORIGIN option is not specified on the LOAD command.

LOADLIB

displays the names of all files, that have a file type of LOADLIB, that are to be searched for load modules (that is, all LOADLIBs specified on the last GLOBAL LOADLIB command, if any).

Response:

```
LOADLIB = libname1 ... libname8
      .           .           .
      .           .           .
      .           .           .
```

Up to eight names are displayed per line, for as many lines as necessary. If no LOADLIBs are to be searched, the following message is displayed at the terminal:

```
LOADLIB = NONE
```

LOCK [*fn ft*] [*dirid*]

displays lock information on a Shared File System (SFS) directory or a file in a directory. Files and directories are locked using the CREATE LOCK command. To remove a lock, use the DELETE LOCK command.

To query a file, specify *fn ft* and the *dirid* containing the file. To query a directory, specify only the *dirid*. If *dirid* is not specified, the directory accessed as A is queried. You must have read or write authority for the file or directory you query and the directory cannot be open. See "Naming Shared File System (SFS) Directories" on page 4 for a description of *dirid*.

Special characters (* or %) can be used for *fn ft* to query a set of files. When querying a set of files, the files and directories that are locked, match the pattern, and for which you have read or write authority are displayed. See "Pattern Matching" on page 8 for more information on using these special characters.

When pattern matching is done on subdirectory names that contain more than eight characters, the first eight characters are used as the file name and the remaining characters are used as the file type. For example, your directory accessed as A contains,

```
CROCKETT NOTEBOOK
CROCKETTNOTES
```

where CROCKETT NOTEBOOK is a file and CROCKETTNOTES is a subdirectory. Issuing,

```
query lock croc* n* a
```

would find both CROCKETT NOTEBOOK and CROCKETTNOTES because CROCKETTNOTES contains more than eight characters and is matched as if it had a file name of CROCKETT and a file type of NOTES. Issuing,

```
query lock crockettnotes * a
```

would also find both CROCKETT NOTEBOOK and CROCKETTNOTES because only the first eight characters are used as the file name. Therefore, any file or alias with the name of CROCKETT, or any subdirectory with CROCKETT as the first eight characters in its name would be listed.

Response:

For QUERY LOCK *fn ft [dirid]*:

```
Directory = filepoolid:userid.n1.n2...n8
Filename Filetype Fm   Type   Userid   Lock   Duration
fn        ft        fm   type   userid   lock   length
.         .         .   .     .        .     .
.         .         .   .     .        .     .
.         .         .   .     .        .     .
```

Note: The second line of the header is generated only if output is displayed at the terminal; the first line containing the directory name is always included.

For QUERY LOCK *dirid*:

```
Directory = filepoolid:userid.n1.n2...n8
Userid    Lock    Duration
userid    lock    length
.         .         .
.         .         .
.         .         .
```

where:

fn
is the name of the file you are querying.

ft
is the file type of the file you are querying.

fm
is the file mode if the directory is accessed, otherwise a dash is displayed.

type
is the type of object locked:

- BASE base file
- ALIAS an alias for a base file
- DIR directory

userid
is the user ID holding the lock.

lock
is the type of lock:

- SHARE others may read while you read
- UPDATE others may read while you read or modify
- EXCLUSIVE others cannot read or write

length
is the duration of the lock:

- LASTING lock remains in effect until a DELETE LOCK command removes the lock
- SESSION lock is removed when your CMS session with the file pool ends (see the CREATE LOCK command for details).

If no locks are in effect, and the STACK, LIFO, or FIFO option was specified, the return code is set to 6, indicating that no data was stacked.

MACLIB

displays the names of all files, with a file type of MACLIB, that are to be searched for macro definitions (that is, all MACLIBs specified on the last GLOBAL MACLIB command, if any).

Response:

```
MACLIB = libname1 ... libname8
      .           .           .
      .           .           .
      .           .           .
```

Up to eight names are displayed per line, for as many lines as necessary. If no macro libraries are to be searched for macro definitions, the response is:

```
MACLIB = NONE
```

NAMEDEF

displays the current namedefs that were established by issuing the CREATE NAMEDEF command.

Response:

Namedef	Filename	Filetype	Directory Name
namedef	-	-	dirname
namedef	fn	ft	-
.	.	.	.
.	.	.	.
.	.	.	.

Note: The header is generated only if output is displayed at the terminal.

where:

namedef

is a temporary name that is associated with either the directory name or the file name and file type.

fn

is the file name associated with the namedef. If a dash is displayed, it means that a directory name is associated with the namedef.

ft

is the file type associated with the namedef. If a dash is displayed, it means that a directory name is associated with the namedef.

dirname

is the complete directory name associated with the namedef. If a dash is displayed, it means that only a file name and file type are associated with the namedef.

If no namedefs are in effect and the STACK, LIFO, or FIFO option was specified, the return code is set to 6, indicating that no data was stacked.

NONDISP

specifies the character that is set to be displayed in place of nondisplayable characters.

Response:

```
NONDISP    char
```

where:

char

is the character that is displayed in place of nondisplayable characters.

OPTION

displays the compiler options that are currently in effect.

Response: OPTION = options...

OUTPUT

displays the contents of any output translate table in effect.

Response:

```
OUTPUT xx1 a1
      . .
      . .
      . .
      xxn an
```

If you do not have an output translate table defined, the response is:

NO USER DEFINED OUTPUT TRANSLATE TABLE IN USE

PROTECT

displays the status of CMS nucleus protection.

Response:

PROTECT = ON

or

PROTECT = OFF

where:

ON

means CMS nucleus protection is in effect.

OFF

means CMS nucleus protection is not in effect.

RDYMSG

displays the format of the CMS ready message.

Response:

RDYMSG = LMSG

or

RDYMSG = SMSG

where:

LMSG

is the standard CMS ready message:

Ready; T = 0.12/0.33 17:06:20

SMSG

is the shortened CMS ready message:

Ready;

REDTYPE

displays the status of the REDTYPE indicator.

Response:

QUERY

REDTYPE = ON

or

REDTYPE = OFF

where:

ON

types CMS error messages in red, for certain terminals equipped with the appropriate terminal feature and a two-color ribbon. Supported terminals are described in the *VM/SP Terminal Reference*.

OFF

does not type CMS error messages in red.

RELPAGE

indicates whether pages of storage are to be released or retained after certain commands complete execution.

Response:

RELPAGE = ON

or

RELPAGE = OFF

where:

ON releases pages.

OFF retains pages.

REMOTE

displays the manner in which data transmission is handled.

Response:

REMOTE ON

or

REMOTE OFF

where:

ON

specifies that data is compressed by removing nulls and combining data when five or more of the same characters occur consecutively in a data stream.

OFF

specifies that the data stream is not compressed. Data is transmitted with no minimization.

SEARCH

displays the search order of accessed disks and Shared File System (SFS) directories.

Response:

label	vdev	mode	stat	dirname
.
.
.
.

Note: If the search order includes any OS or DOS disks, the response will also indicate whether the disk is OS or DOS.

where:

label

is either the label assigned to the disk when it was formatted or the volume label if it is an OS or DOS disk. If it is an SFS directory, a dash is displayed in the label column.

vdev

is either the virtual device address for a disk or "DIR" for an SFS directory.

mode

is the file mode letter assigned to the disk or SFS directory when it was accessed. Mode may also be indicated as an extension of a disk or directory, such as C/A. However, when you use QUERY SEARCH (different than QUERY DISK), you will not see any extension indicated if it is an extension of itself. For example, for C/C, you will see only C.

stat

indicates whether the status of the disk or SFS directory is read/write (R/W) or read/only (R/O).

dirname

is the complete name of an SFS directory. This column is left blank for disks. In a full-screen CMS environment if the directory name is too long to display on the line, you can scroll to the right to see the remainder of the directory name.

SEGMENT *segname* CONTENTS

returns a list of the objects in the specified logical segment space.

Response:

Type	Location	Length	Name
type	location	length	name
.	.	.	.
.	.	.	.
.	.	.	.

where:

type

indicates what the object is. For example, the object might be an exec, CSL routine, or LANGUAGE.

location

indicates the starting address of the object in the segment.

length

is the length of the object in the segment.

name

is the name of the object in the segment.

Example:

Entering the command:

```
query segment myseg contents
```

would list the contents of the logical segment named MYSEG,

QUERY

Type	Location	Length	Name
NUCEXT	006E0630	00000038	TESTMOD1
SUBCOM	006E0F18	00000038	TESTMOD2
EXEC	006E32D0	00000848	PROF1 EXEC
EXEC	006E0698	00000848	TEST XEDIT
LANGUAGE	006E3030		AMENG OFS
CSL	006E0000	00000610	LIB2
USER	006E3B50	000000FF	TESTUSER

If MYSEG contained a single shared minidisk directory the output might look like this:

Type	Location	Length	Name
DISK	006E5000	00010000	LABEL1

SEGMENT *segname* ASSIGN

returns the name of the physical segment that the specified logical segment is assigned to.

Response:

Lsegname Assignment
lsegname assignment

where:

lsegname
is the name of the specified logical segment.

assignment
is the name of the physical segment that the logical segment is assigned to.

SEGMENT *segname* SPACE

returns information about the segment space containing the specified segment.

Response:

Space	Name	Location	Length	Loaded	Attribute
space	-	location	length	YES/NO	-

where:

space
is the name of the physical segment or segment space

location
is the starting address of the segment space.

length
is the length of the segment space.

SEGMENT * SPACE

returns information about all the physical segments and segment spaces that contain reserved or loaded segments.

Response:

The response is the same as for QUERY SEGMENT *segname* SPACE; one line is returned for each segment space.

SEGMENT *segname* [PHysical]LOGical]

returns information about the specified physical or logical segment.

Response:

Space	Name	Location	Length	Loaded	Attribute
space	segname	location	length	YES/NO	SYSTEM/USER

where:

space

is the name of the physical segment or segment space.

segname

is the name of the saved segment associated with the segment space. For logical segments this will be the logical segment name. For physical segments this will be the name of the reserved segment space.

location

is the starting address of the segment space.

length

is the length of the segment space.

The Loaded column indicates whether or not the segment is attached to a virtual machine. If SYSTEM is returned in the Attribute column it means that the space will survive abend processing. If USER is returned it indicates that the space will not survive abend processing.

SEGMENT * [PHysical]LOGical]

returns information about all physical and/or logical segment spaces defined by the SEGMENT command and the SEGMENT macro. This is the default.

Response:

The response is the same as for QUERY SEGMENT *segname* [PHYSICAL]LOGICAL]; one line is returned for each segment space.

SERVER

displays whether the virtual machine will process or reject private resource conversation requests.

Response:

SERVER	ON
	OFF

where:

ON

indicates the virtual machine will process any private resource conversation requests.

OFF

indicates the virtual machine will reject any private resource conversation requests.

STORECLR

displays the current setting of CMS GETMAIN free storage clean up.

Note: Use the SET STORECLR command to change the way CMS releases GETMAIN storage.

Responses:

STORECLR = ENDSVC

or

STORECLR = ENDCMD

where:

ENDSVC

indicates that CMS releases GETMAIN storage when the program that called GETMAIN issues an SVC to return to the program that called it. CMS treats the STRINIT macro as a NOP.

ENDCMD

indicates that CMS returns GETMAIN storage at end-of-command, the point where the CMS ready message is displayed. CMS honors user invocations of STRINIT.

SYNONYM SYSTEM

displays the CMS system synonyms in effect that are enabled by the SYNONYM command.

Response:

SYSTEM	SHORTEST
COMMAND	FORM
-----	-----
command	minimum truncation
.	.
.	.
.	.

If no system synonyms are in effect, the following message is displayed at the terminal:

No system synonyms in effect

SYNONYM USER

displays user synonyms in effect that are enabled by the SYNONYM command.

Response:

SYSTEM	USER	SHORTEST
COMMAND	SYNONYM	FORM (IF ANY)
-----	-----	-----
command	synonym	minimum truncation
.	.	.
.	.	.
.	.	.

If no user synonyms are in effect, the following message is displayed at the terminal:

No user synonyms in effect

SYNONYM ALL

displays all synonyms in effect that are enabled by the SYNONYM command.

Response: The response to the command QUERY SYNONYM SYSTEM is followed by the response to QUERY SYNONYM USER.

SYSNAMES

displays the names of the standard saved systems.

Response:

SYSNAMES: CMSVSAM CMSAMS CMSDOS CMSBAM

ENTRIES: entry... entry... entry... entry...

where:

SYSNAMES are the standard names that identify the discontinuous saved systems.

ENTRIES are the standard system default names or the system names established via the SET SYSNAME command.

TEXT

displays the status of TEXT character code conversion.

Response:

TEXT ON

or

TEXT OFF

where:

ON

converts TEXT characters for windows.

OFF

does not convert TEXT characters.

TRANslate

displays translations and translation synonyms that are in effect.

SYStem

displays only the System National Language Translation Table.

USER

displays only the User National Language Translation Table.

ALL

displays System and User National Language Translation Tables.

TRANslate

displays only the national language translations.

SYNonym

displays only the national language translation synonyms.

BOTH

displays both the national language translations and translation synonyms.

APPLid *applid*

is an application identifier that defines information about a particular application. It must be three alphanumeric characters, and the first character must be alphabetic. The default, *, displays tables for all applications.

Response:

The QUERY TRANSLATE command responds with:

- One or two messages stating whether or not the translations and translation synonyms are active, and
- A table displaying the command name, translations/synonyms, and minimum abbreviation, or a message stating that there are no entries in the table.

In the first part of the response, you receive one or more of the following messages, depending on the options that you specify:

System translations, application id: *applid*

A heading for the currently active system translations

System translation synonyms, application id: *applid*

A heading for the currently active system translation synonyms

User translations, application id: *applid*

A heading for the currently active user translations

User translation synonyms, application id: *applid*

A heading for the currently active user translation synonyms

User translations off, application id: *applid*

A response when user translations are currently not active

User translation synonyms off, application id: *applid*

A response when user translation synonyms are currently not active

System translation off, application id: *applid*

A response when system translations are currently not active

System translation synonyms off, application id: *applid*

A response when system translation synonyms are currently not active

No entries in table

A response when there are no entries in table being displayed

When the tables contain entries for translations and translation synonyms, the tables are displayed in the following order:

- User Translations
- System Translations
- User Translation Synonyms
- System Translation Synonyms

Each table is displayed in the following format:

command name	translation/synonym	minimum	abbreviation
.	.	.	.
.	.	.	.
.	.	.	.

Example:

To display only the system translations that are in effect, enter
 query translate system translate

TXTLIB

displays the names of all files, with a file type of TXTLIB, that are to be searched for unresolved references (that is, all TXTLIBs specified on the last GLOBAL TXTLIB command, if any).

Response:

TXTLIB = libname1 ... libname8
.
.
.

Up to eight names are displayed per line, for as many lines as necessary. If no TXTLIBs are to be searched for unresolved references, the following message is displayed at the terminal:

TXTLIB = NONE

UPSI

displays the current setting of the UPSI byte. The eight individual bits are displayed as zeros or ones depending upon whether the corresponding bit is on or off.

Response: UPSI = nnnnnnnn

Options

STACK

causes the results of the QUERY command to be placed in the program stack instead of being displayed at the terminal. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

If CMS passes the command to CP, then the response from CP is also put in the program stack. If CP precedes the QUERY command, CMS does not stack the results. The STACK option is valid only when issued from CMS. Error messages are displayed at the terminal and are not stacked.

FIFO

(first-in first-out) is the default option for STACK. FIFO causes the results of the QUERY command to be placed in the program stack instead of being displayed at the terminal. The information is stacked FIFO. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

(last-in first-out) causes the results of the QUERY command to be placed in the program stack rather than being displayed at the terminal. The information is stacked LIFO. This option is equivalent to STACK LIFO.

XEDIT

places the information in the file that is currently displayed. This option is only valid when issued from the XEDIT environment. The output replaces the information in the file starting with the current line and continues until the end of the output is reached. Remaining text in the file, if any, is unchanged. The edited file must either be variable length format or fixed format with a logical record length (lrecl) of 190 for QUERY ALIAS or a lrecl of 165 for QUERY AUTHORITY.

The XEDIT option of the QUERY command can only be used with QUERY ALIAS and QUERY AUTHORITY.

Usage Notes

1. You may specify only one QUERY parameter at a time.
2. If the implied CP (IMPCP) function is in effect and you enter an invalid QUERY parameter, you may receive the message DMKCCQG045E - userid NOT LOGGED ON.
3. If an invalid QUERY parameter is specified from an exec and the implied CP (IMPCP) function is in effect, then the return code is -0003.
4. The DOSPART, OPTION, and UPSI functions are valid only if the CMS/DOS environment is active.

QUERY

5. If the information that you query requires a response from CP and the response is longer than 8192 characters, nothing is stacked and you receive a return code of 88.
6. The language for QUERY command responses depends on the language used to enter the command and whether or not the command was translated. If the QUERY command is entered in American English (or AMENG, which is always available), CMS responds in American English. If the command is entered in the current national language, or if the translation of the command is the same as American English, the response is displayed (or stacked) in the current national language. The language of the response is especially important for language dependent execs. For more information, see the section "Using Translations" in the *VM/SP CMS User's Guide*.

Messages and Return Codes

DMSJNL109S	Virtual storage capacity exceeded [RC = 104]
DMSJNL637E	Missing nodeid for the AT operand [RC = 24]
DMSJNL647E	Userid not specified for <i>nickname</i> in <i>userid</i> NAMES file [RC = 32]
DMSJNL647E	Localid not specified for <i>userid</i> at <i>node</i> in <i>userid</i> NAMES file [RC = 32]
DMSJNL653E	Error executing <i>command</i> rc= <i>nn</i> [RC = 40]
DMSOUT105S	Error <i>nn</i> writing file to XEDIT [RC = 100]
DMSOUT688E	XEDIT option only valid from XEDIT environment [RC = 24]
DMSQRE386E	Missing operand(s) [RC = 24]
DMSQRE389E	Invalid filepoolid: <i>filepoolid</i> [RC = 24]
DMSQRE391E	Unexpected operand(s): <i>operand</i> [RC = 24]
DMSQRE1202E	Userid must not be specified if {ALL *} is specified [RC = 24]
DMSQRE1223E	There is no default file pool currently defined [RC = 40]
DMSQRE1238E	Missing <i>userid</i> for <i>operand</i> operand [RC = 24]
DMSQRF926E	Command is only valid on a display terminal [RC = 88]
DMSQRN1222I	No user defined NAMEDEF in effect
DMSQRP065E	<i>option</i> option specified twice [RC = 24]
DMSQRP689E	File must be F-format <i>nnn</i> or V-format [RC = 24]
DMSQRQ1239E	You are not authorized to issue this request on behalf of <i>userid</i> [RC = 76]
DMSQRQ1239E	You are not authorized to issue this request for ALL users [RC = 76]
DMSQRR002E	File <i>fn ft fm dirname</i> not found [RC = 28]
DMSQRR109E	Virtual storage capacity exceeded [RC = 109]
DMSQRR1160E	Directory <i>dirname</i> already open [RC = 70]
DMSQRR1184E	File <i>fn ft fm dirname</i> not found or you are not authorized for it [RC = 28]
DMSQRR1184E	Directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSQRR1184E	File <i>fn ft</i> or directory <i>dirname</i> not found or you are not authorized for it [RC = 28]
DMSQRR1184E	File <i>fn ft</i> or directory <i>dirid</i> not found [RC = 28]
DMSQRR1185I	No locks are held on <i>fn ft fm dirname</i>
DMSQRR1185I	No locks are held on directory <i>dirname</i>
DMSQRR1186I	No alias exists for <i>fn ft fm dirname</i>
DMSQRR1210E	Directory <i>dirname</i> not found [RC = 28]
DMSQRS104S	Error <i>nn</i> reading SYSTEM LANGUAGE S from disk [RC = 1nn]
DMSQRS280E	Application <i>applid</i> not active [RC = 28]
DMSQRX070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSQRX099E	CMS/DOS environment [not] active [RC = 40]
DMSQRY003E	Invalid option: <i>option</i> [RC = 24]

DMSQRY014E Invalid function *function* [RC=24]
 DMSQRY066E *option1* and *option2* are conflicting options [RC=24]
 DMSQRY618E NUCEXT failed [RC=*m*]
 DMSQRY621E Bad plist: *message* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

RDR

Use RDR to determine the characteristics of the next file in your virtual reader. RDR generates a return code and either displays or stacks a message for each type of file recognized. Which file is "next" depends upon the class of the reader, the class of the files in the reader, and whether or not they are held.

Format

RDR	$\left[\begin{array}{c} \text{spool-class} \\ = \end{array} \right] \quad \left[(\text{options... } []) \right]$
	<p><u>Options:</u></p> $\left[\begin{array}{l} \text{NOTYPE} \\ \text{STACK} \quad \left[\begin{array}{c} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{FIFO} \\ \text{LIFO} \end{array} \right] \quad \left[\begin{array}{l} \text{MSGSUBS} \\ \text{MSGALL} \end{array} \right]$

Operands

spool-class

is the class of the spool file for which information is to be returned. The virtual reader remains spooled to the class specified in the RDR command.

=

indicates that information is to be returned for a file having the same spool file class as that of the virtual reader. This is the default.

Options

NOTYPE

specifies that no message is to be displayed or stacked. However, a return code is generated, which is accessible from within an EXEC 2 (or EXEC) procedure, by examining the variable &RC (or &RETCODE).

STACK [FIFO]

STACK LIFO

specifies that the message is placed in the program stack rather than displayed at the terminal. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO

specifies that the information is placed in the program stack rather than displayed at the terminal. The information is stacked FIFO. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

specifies that the information is placed in the program stack rather than displayed at the terminal. The information is stacked LIFO. This option is equivalent to STACK LIFO.

MSGSUBS

returns only the available substitution information for the current spool file. The lines are displayed or stacked in accordance with the NOTYPE, STACK, LIFO, or FIFO options described above. See the usage notes for additional information on substitution data.

MSGALL

returns the normal message and all available substitution information for the current spool file. The lines are displayed or stacked in accordance with the NOTYPE, STACK, LIFO, and FIFO options described above. See the usage notes for additional information on substitution data.

Usage Notes

1. Issued with no options, RDR displays the return code and message.
2. Issued with the NOTYPE option from an exec (written in the REXX language) RDR places a return code in the variable RC. Appropriate action can be taken by examining this variable. For example:

```
rdr '('notype
If rc=22 Then disk load
Else If RC=7 Then readcard
```

3. If the spool-class specified is different from the current spool class of the virtual reader, the virtual reader's spool class is changed to the one specified. The current spool class of the virtual reader can be determined by issuing the CP command QUERY VIRTUAL 00C or QUERY VIRTUAL UR.
4. The RDR command changes the order of the files in your virtual reader. Files that are not held are re-ordered according to class.
5. The RDR command does not handle MONITOR files.

Responses

The return codes and messages are:

```
0   Reader empty
1   System dump file
2   PRINTER FILE (ITEM LENGTH lrecl)
3   DISK LOAD fn ft fm
4   :READ fn ft fm originid mm/dd/yy hh:mm:ss
5   Cards for IPL
6   Unnamed card deck
7   :READ fn ft fm
9   Reader not operational
10  File with X'5A' CCW
13  Reader not ready
18  Console spool file
22  DISK LOAD fn ft fm
23  Netdata file
26  Message
```

Explanations of the messages follow. (The return code is not part of the message.)

Reader empty RC=0

The reader is empty, the reader file is held, or there are no files in the reader of the current reader spool class. You can check to make sure the reader corresponds to the current spool class, or check for held files.

System dump file RC = 1

The reader contains a system dump file, which can be handled using the appropriate system utility.

PRINTER FILE (ITEM LENGTH *lrecl*) RC = 2

The reader contains a printer file with a logical record length of *lrecl*.

DISK LOAD *fn ft fm* RC = 3

The reader contains a file sent via DISK DUMP from a VM/370 Release 6 (or earlier version) CMS file system. The CMS file system in VM/370 Release 6 (or earlier) only supports minidisks formatted in 800-byte physical blocks.

:READ *fn ft fm originid mm/dd/yy hh:mm:ss* RC = 4

The reader contains a non-console or a non-printer file that has a READ control card as the first real record. If the PUNCH command produced this file, then the result will be in the above format; otherwise, the READ control card will be displayed through column 72.

Cards for IPL RC = 5

The reader contains a file that has IPL cards as the first cards in the file.

Unnamed card deck RC = 6

The reader contains a PUNCH file that can not be identified.

:READ *fn ft fm* RC = 7

The reader contains a file that is a printer file.

Reader not operational RC = 9

The reader is not operational: device 00C does not exist in the virtual machine configuration or device 00C is not a reader. Possible causes: the reader is not defined in your CP directory entry; the reader was detached; some other device was at 00C. (CMS requires the reader to be at address 00C.)

PRINTER FILE (ITEM LENGTH *lrecl*) RC = 10

The reader contains a printer file with a X'5A' CCW. The CCW was created by CP from a carriage control character of X'5A'. Files containing this carriage control character are intended for an all points addressable printer.

Reader not ready RC = 13

The reader is not ready. To reverse the not ready status, issue the CP command READY 00C.

Console spool file RC = 18

The reader contains a file that is from a console.

DISK LOAD *fn ft fm* RC = 22

The reader contains a file sent via DISK DUMP from a post-VM/370 Release 6 CMS file system or from the Shared File System (SFS). In addition to the 800-byte physical blocksize used by the VM/370 file system, the enhanced file system supports minidisks formatted in 512-, 1024-, 2048-, or 4096-byte logical blocks.

Netdata file RC = 23

The reader contains a file that was sent using the SENDFILE command with the NEW option.

Message RC = 26

The reader contains non-console or non-printer file that has a MSG control card as the first real record.

If RDR is issued with the MSGSUBS or MSGALL option the RDR command uses the following response format when returning the substitution data that is available. This response has a fixed format and fixed length. The length, including blank delimiters, is 98 characters.

The format is (each field is delimited by one blank):

```
msgid fn      ft      fm lrl xxxx
```

where:

msgid

specifies the message identifier, issued with all return codes, and has a length of six

fn

specifies the file name, issued with a return code of 3 or 22, and has a length of eight

ft

specifies the file type, issued with a return code of 3 or 22, and has a length of eight

fm

specifies the file mode, issued with a return code of 3 or 22, and has a length of two

lrl

specifies the logical record length, issued with a return code of 2, and has a length of three

xxxx

specifies header record text, issued with a return code of 4 or 7, and has a length of 66

Example: The following example shows the response that you will receive when you issue RDR (MSGSUBS against a printer file that is in your reader. This generates a return code of 2.

```
816501 *      *      * 132 *
```

- Each individual field in the substitution line is initialized to an '*' followed by enough blanks to fill the field, except the LRECL field which is numeric and initialized to zero.
- The first piece of substitution information is the message identifier that consists of a four digit message number ('8165') concatenated with a two digit format number ('01'). It is the actual message number of the message issued at the completion of the RDR command.
- '132' is the only substitution data available. '132' is the logical record length (LRECL) of a printer file that is in your reader.

If RDR is issued with the MSGSUBS or MSGALL option and stacking options are not specified, the results are returned to the terminal.

- If MSGSUBS is specified, only the message identifier and the available substitution data is returned. The message itself is not stacked or returned to the terminal.
- If MSGALL is specified, both the actual message and the message identifier with the available substitution data are returned. The actual message is the first line returned, and the substitution data is the second line returned.

Example: The following example assumes there is a printer file in your reader.

- If you issue RDR (MSGSUBS, the following is returned to the terminal:
816501 * * * 132 *

If RDR is issued with the MSGSUBS or MSGALL option and stacking options are specified, the results are returned in the program stack.

Examples: The following examples assume there is a printer file in your reader.

- If you issue RDR (STACK MSGALL, the following is returned in the stack:
PRINTER FILE (ITEM LENGTH 132)
816501 * * * 132 *

If you issue a PULL from within a REXX EXEC, you will receive the actual message from the first line in the stack, and the message identifier with the substitution data from the second line.

- If you issue RDR (LIFO MSGALL, the following is returned in the stack:
816501 * * * 132 *
PRINTER FILE (ITEM LENGTH 132)

If you issue a PULL from within a REXX EXEC, you will receive the message identifier and the substitution data from the first line in the stack, and the actual message from the second line.

Messages and Return Codes

DMSRDR070E Invalid parameter *parameter* [RC = 24]
DMSRDR630S Error accessing spool file [RC = 36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

RDRLIST

Use the RDRLIST command to display information about the files in your virtual reader. The RDRLIST environment is controlled by the System Product Editor. Therefore, you can use XEDIT subcommands to manipulate the files. In addition, you can look at a given reader file, discard it, copy it to a CMS disk or directory, or transfer it to someone else (local or remote).

In most cases these files were sent to you by other computer users, on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network.

Format

RDRList RList	[(options... [])]
	<u>Options:</u> [PROFile <i>fn</i>] [Append]

Options

PROFile *fn*

specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the RDRLIST command. If not specified, the default macro PROFRLST XEDIT is invoked. For more information on the PROFRLST macro, see the usage note, "Default PF Key Settings."

Append

specifies that the list of files in your reader should be appended to the existing list. This option has meaning only when issued from within RDRLIST, and is ignored otherwise.

Usage Notes

1. You can use the special commands EXECUTE and DISCARD from the RDRLIST screen. The EXECUTE command allows you to issue commands that use the reader files displayed by RDRLIST. See "EXECUTE" on page 794 for more information. The DISCARD command allows you to purge the reader files displayed by RDRLIST. See "DISCARD" on page 792 for more information.
2. Tailoring the RDRLIST Command Options
 You can use the DEFAULTS command to set up options and/or override command defaults for RDRLIST. However, the options you specify in the command line when entering the RDRLIST command override those specified in the DEFAULTS command. This allows you to customize the defaults of the RDRLIST command, yet override them when you desire. Refer to the DEFAULTS command description for more information.
3. Format of the List

When you invoke the RDRLIST command you are placed in the XEDIT environment, editing a file "userid RDRLIST A1." The existing copy of this file is erased if it exists.

The file you are editing is a list of files with information collected from the CP QUERY RDR ALL command. Each line contains:

- a command area
- file name and file type
- class and type
- number of records
- whether or not the file is held
- creation date and time
- originating user ID and node

The full power of XEDIT is available to you while you issue commands against the list of files. For example, you may want to use XEDIT subcommands to scroll through the list of files, locate a particular file, etc.

However, some XEDIT subcommands are inappropriate in this environment. Subcommands that alter the format or the contents of "userid RDRLIST" (for example, SET TRUNC, SET FTYPE, or SET LINEND) may cause unpredictable results.

4. Entering CMS commands from RDRLIST:

Begin CMS commands with "CMS" to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.

5. Issuing Commands from the List

On a full screen display, you can issue commands directly from the line on which a reader file is displayed. These commands must be CP or CMS commands that operate on reader files (for example, CHANGE RDR, PURGE RDR, TRANSFER RDR, PEEK, DISCARD). For the above commands that operate on the reader files, the spoolid number is automatically appended to the end of the command. Use the slash (/) symbols described below to specify the spoolid elsewhere in the command. For example:

```
CHANGE RDR / CLASS A
RECEIVE / fn ft ( REPLACE
```

To enter a command, just move the cursor to the line that describes the file to be used by the command, and type the command in the space provided to the left of the file name. If a command is longer than the command space provided on the screen, just continue typing over the rest of the line. You press the ENTER key to execute the command. (The ENTER key is set to EXECUTE, which is described in the section "EXECUTE" on page 794.

For example, to transfer a reader file to a user on your computer (local), you would move the cursor to that line on the screen and type:

```
TRANSFER RDR / USERA
```

where USERA is the user ID of the recipient. To transfer a reader file to a user on another computer that is connected to yours (remote), you must know the user ID of the user, the user ID (rscsid) of the virtual machine at your location that is running the Remote Spooling Communications Subsystem (RSCS), and

the location identification (locid) of the computer at the remote location. Move the cursor to that line on the screen and type:

```
TAG FILE / REMOTE1 USERB
```

where REMOTE1 is the locid of the remote computer and USERB is the user ID of the recipient. After you have entered the TAG command, on the same line type:

```
TRANSFER RDR / NET
```

where NET is the user ID (rscsid) of the virtual machine at your location that is running RSCS.

To purge a file, you would move the cursor up to that line on the screen, and type "discard" in the space provided to the left of the file name. DISCARD is another special command described in "DISCARD" on page 792. When the ENTER key is pressed, all the commands typed on one screen are executed. The screen is restored to its previous state; however, the list is updated to reflect the current status of the files (see "Responses").

You may want to enter commands from the RDRLIST command line before executing commands that are typed on the list. To do this, move the cursor to the command line by using the PF12 key (instead of the ENTER key). After typing a command on the command line and pressing ENTER, you can use PF12 to move the cursor back to its previous position on the list.

Another way to issue commands that make use of the reader files displayed is to issue EXECUTE from the RDRLIST command line. A complete description of EXECUTE is on page 794.

6. Using Symbols as Part of a Command

Symbols can be used to represent operands in the command to be executed. They can be used in the commands typed on the list, or as part of the command in EXECUTE (on the command line). Symbols are needed if the command to be executed has operands or options that follow the command name. Examples of using symbols are in the "Examples" section, below. The following symbols can be used:

/ means the spoolid of the file displayed on the line.
 /n means the file name displayed on the line.
 /t means the file type displayed on the line.
 /o means execute the line as is, without appending anything.
 /m means the device type (from which the file was sent).

Any combinations of symbols can be used. For example:

/n /t means: file name followed by file type.
 /nt means: file name followed by file type.

Note: If the symbol '/' appears in a command or in its operands, it must be issued from the command line, and not as part of an EXECUTE command.

7. Special Symbols Used Alone

The following special symbols can be typed alone on the lines of the RDRLIST display. They have the following meanings:

- = means execute the previous command for this file. Commands are executed starting at the top of the screen. For example, suppose you enter DISCARD on a line. You can then type an equal sign on any other line(s) below the DISCARD command. Those files preceded by equal signs are discarded when the EXECUTE command is entered (from the command line or by pressing the ENTER key).
- ? means display the last command executed. The command is displayed on the line in which the ? is entered.
- / means make this line the current line. (On the RDRLIST screen, the current line is the first file on the screen.)

8. Default Key Settings and Synonyms

The PROFRLST XEDIT macro is executed when the RDRLIST command is invoked, unless you specified a different macro in the RDRLIST command. It sets the keys to the following values:

ENTER		Execute commands typed in the file line(s) or on the command line. (The ENTER key is set by the XEDIT subcommand, SET ENTER IGNORE MACRO EXECUTE).
PF 1	Help	Display RDRLIST command description.
PF 2	Refresh	Update the list to indicate discarded files, etc.
PF 3	Quit	Exit from RDRLIST display.
PF 4	Sort	by file type, file name.
PF 5	Sort	by date and time, in a calendar year, oldest to newest.
PF 6	Sort	by user ID, in alphabetical order.
PF 7	Backward	Scroll back one screen.
PF 8	Forward	Scroll forward one screen.
PF 9	Receive	Receive the file pointed to by the cursor (see the RECEIVE command).
PF 10		Not assigned
PF 11	Peek	Display file where cursor is placed, but do not write it on a disk or directory. The file is displayed in the XEDIT environment. See also the PEEK command description.
PF 12	Cursor	If cursor is in the file area, move it to the command line; if cursor is on the command line, move it back to its previous location in the file (or to the current line).

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

Some XEDIT subcommands are stacked by the FILELIST command (for example, SET TRUNC, SET LRECL, and SET VERIFY). In order to override these settings in a profile, these SET subcommands must be stacked FIFO.

In addition to setting the above PF keys, the PROFRLST XEDIT macro sets the synonyms that sort your RDRLIST files. Enter the synonyms on the RDRLIST command line. The synonyms and their descriptions are:

SNAME Sorts the list alphabetically by spool file name and spool file type.

STYPE Sorts the list alphabetically by spool file type and spool file name.

- SCLAS** Sorts the list by device type, class, and hold status.
- SHOLD** Sorts the list by hold status, device type, and class.
- SUSER** Sorts the list by origin user ID, node, and date.
- SSIZE** Sorts the list by the number of records (greatest to least).
- SDATE** Sorts the list month, day, and time (oldest to most recent).

9. Displaying a File

To display a file on the screen without reading it onto a disk or directory position the cursor at the file you want to see and press the PF11 key, which is set to the PEEK command. Refer to the PEEK command for more information on the PEEK screen.

10. If you want to issue RDRLIST from an exec program, you should precede it with the EXEC command; that is, specify


```
exec rdrlist
```
11. You may receive a response on the RDRLIST screen on the line where the reader file is listed. To issue commands on that line, type over the response, press the ERASE EOF key (or space over the rest of the line) and press ENTER. Alternatively, press the PF2 key (REFRESH), move your cursor to the appropriate line, and enter the command.
12. When you press PF9 (RECEIVE), you may receive prompting messages that require a response. See the RECEIVE command for an explanation of the prompts and responses.

Examples

In the RDRLIST environment, information about the user's virtual reader is displayed in a format similar to what the FILELIST command provides about a CMS disk or Shared File System (SFS) directory.

The following is a sample RDRLIST screen.

```

OHARA  RDRLIST  A0  V 108  Trunc=108  Size=17  Line=1  Col=1  Alt=1
Cmd    Filename Filetype Class User At Node Hold  Records  Date  Time
PIZZA  TOPPINGS PUN  A  KEN  NODE04 NONE    10  10/06 10:39:38
COOKIE ASSEMBLE  PUN  A  KEN  NODE04 NONE    10  10/06 10:25:11
$JELLY SCRIPT  PRT  A  KEN  NODE04 NONE     7  10/06 10:15:50
DIETING TIPS    PUN  A  KEN  NODE04 NONE    11  10/06 09:40:28
KEN    NOTE    PUN  A  KEN  NODE04 NONE    10  10/06 08:43:07
SEND   EXEC    PUN  A  BOB  NODE02 NONE     2  10/06 07:12:35
GOOD   DAY     PUN  A  GEOFF NODE02 NONE    29  10/05 11:44:34
Acknowl edgment PUN  A  BOB  NODE02 NONE     2  10/05 11:42:21

1= Help    2= Refresh 3= Quit    4= Sort(type) 5= Sort(date) 6= Sort(user)
7= Backward 8= Forward 9= Receive 10=          11= Peek     12= Cursor

====>

X E D I T  1 File
    
```

Figure 19. Sample RDRLIST Screen

The following examples show how symbols can be used to represent operands in a command. The values substituted for the symbols and the resulting command are shown. In each case, the command can be entered in either of the following ways:

- typed in the “Cmd” area of the screen. The command is executed either by pressing the ENTER key or by entering EXECUTE on the XEDIT command line and then pressing ENTER.
- entered from the XEDIT command line, as an operand of EXECUTE (in the form “EXECUTE lines command”).

If a symbol is not specified, the spoolid number of the reader file is appended automatically to the command.

SPOOL FILE ID	COMMAND	RESULTING COMMAND
pizza toppings	DISCARD	DISCARD spoolid
cookie assemble	RECEIVE / CAKE /t (REPLACE	RECEIVE spoolid CAKE ASSEMBLE (REPLACE
ken note	PEEK	PEEK spoolid
send exec	FILELIST /n * *	FILELIST SEND * *
\$jelly script	TRANSFER RDR / TO * PRT	TRANSFER RDR spoolid TO * PRT (prints the file)

Responses

After a command is executed, one of the following symbols is displayed in the "Cmd" space to the left of the file for which it was executed.

- * Means the command was executed successfully (RC=0).
- *n Is the return code from the command executed (RC=n).
- *? Means the command was an unknown CP/CMS command (RC=-3).
- *! Means the command was not valid in CMS subset. For a list of commands valid in CMS subset mode, see the *VM/SP CMS User's Guide*.

The following responses can also appear on the RDRLIST screen after you have issued a command for a file from your virtual reader:

- * spoolfn spoolft ** Discarded or Received **
- * spoolfn spoolft has been discarded.
- File spoolfn spoolft has been discarded.
- * spoolfn spoolft has been left in your reader.

The following response can also appear if RDRLIST is invoked in the CMS environment and you have no reader files:

No files in your reader.

Messages and Return Codes

DMSWRL205E No files in your reader [RC = 28]

DMSWRL651E APPEND must be issued from RDRLIST or FILELIST [RC = 40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

READCARD

Use the READCARD command to read data records from your virtual reader and to create CMS files containing the data records.

Format

READcard	$\left\{ \begin{array}{l} fn \ ft \ \left[\begin{array}{c} fm \\ \underline{A} \end{array} \right] \\ * \ \left[\begin{array}{c} * \\ \underline{A} \end{array} \right] \ \left[\begin{array}{c} fm \\ \underline{A} \end{array} \right] \end{array} \right\} \quad [\text{(options... [])}]$ <p>Options: $\left[\begin{array}{c} \text{Fullprompt} \\ \text{Minprompt} \\ \text{NOPrompt} \end{array} \right]$ $\left[\begin{array}{c} \text{Replace} \\ \text{NOReplace} \end{array} \right]$</p>
-----------------	---

Operands

- fn*
is the file name you want to assign to the file being read.
- ft*
is the file type you want to assign to the file being read.
- * $\left[\begin{array}{c} * \\ \underline{A} \end{array} \right]$
indicates that file identifiers are to be assigned according to READ control cards in the input deck.
- fm*
is the file mode letter of the disk or directory onto which the file is to be read and the file mode number of the file. A file mode of A1 is assumed if this field is omitted or specified as an asterisk (*) on the command. However, if a file mode number is on the control card, that number is used with A. When a file mode letter is specified on the command, the default file mode number is 1, unless the file mode number is on the control card. Specifying a file mode letter and number on the command ignores the file mode letter and number on the control card and assigns that file mode to the file.

Options

- Fullprompt**
specifies that a prompt is issued for each file.
- Minprompt**
specifies that a prompt is issued when the name of the first (or only) file differs from the name of the spool file; the prompt for the first file is suppressed when it has the same name as the spool file. A prompt is always issued for the second and subsequent files. This will only occur if READCARD * is specified. MINPROMPT is the default.
- NOPrompt**
specifies that a prompt is not issued to you as a file is received.

Replace

specifies that if a file of the same file name and file type exists on the disk or in the directory onto which the incoming file is to be loaded, it is to be replaced with this one.

NOREplace

specifies that a file is not received that would overlay an existing file on the receiving disk or directory. NOREPLACE is the default.

Usage Notes

1. Data records read by the READCARD command must be fixed-length records, and may be a minimum of 80 and a maximum of 204 characters.

2. Tailoring the READCARD Command Options

You can use the DEFAULTS command to set up options and/or override command defaults for READCARD. However, the options you specify in the command line when entering the READCARD command override those specified in the DEFAULTS command. This allows you to customize the defaults of the READCARD command, yet override them when you desire. Refer to the DEFAULTS command description for more information.

3. You cannot receive multiple files as one file by spooling your reader continuous (CONT). The READCARD command resets the continuous spooling option and spools your reader NOCONT.

4. If you specify the FULLPROMPT or MINPROMPT option the valid responses include:

- One of the digits specified in the prompt
- One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	If this file is one of a set of files that constitutes a single spool file, the file is not received and prompting continues for the next file, if there is one. If this is the last file of a set of files or if this is the only file in the spool file, the command is ended.
1 or Yes	Receives the file under the name <i>fn1 ft1 fm1</i> (or <i>fn3 ft3 fm3</i>).
2 or Quit	Ends the command.
3 or Rename	Requests prompt message DMSRDC1080R so that the incoming file can be received using a different name.

5. If you receive prompt message DMSRDC1081R the valid responses include:

- One of the digits specified in the prompt
- One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	Does not receive the file under the name <i>fn ft fm</i> and repeats the original prompt message DMSRDC1080R. This allows you to specify a different name for the incoming file.

READCARD

1 or Yes Receives the file under the name *fn ft fm*.

2 or Quit Ends the command.

6. CMS file identifiers are assigned according to READ control cards in the input deck (the PUNCH command header card is a valid READ control card). When you enter the command:

```
readcard *
```

CMS reads the first spool reader file in the queue and if there are READ control cards in the input stream, it names the files as indicated on the control cards.

The first card in the deck may not be a READ control card. If not, CMS writes a file named READCARD CMSUT1 A1 to contain the data, until a READ control card is encountered or until the end-of-file is reached.

7. If you specify a file name and file type on the READCARD command, for example:

```
readcard junk file
```

CMS does not check the input stream for READ control cards, but reads the entire spool file onto the disk or directory and assigns it the specified file name and file type.

If there were any READ control cards in the deck, they are not removed.

Delete them using the editor if you do not want them in your file. If the file is too large, you can either increase the size of your virtual storage (using the CP DEFINE command), or use the COPYFILE command to copy all records except the READ control cards (using the FROM and FOR options).

8. READCARD loads a file from the reader into a temporary work file called "READCARD CMSUT2." The existing file with the same name as the one being loaded from the reader is then erased. The name of the temporary work file just created is changed to the name of the file just received. However, if the file you are loading has the name "READCARD CMSUT2," it will be changed to "READCARD CMSUT3." "READCARD CMSUT2" is a reserved work file name of the READCARD command.
9. To read a file onto a disk or directory other than your disk or directory accessed as A, specify the file mode letter when you specify the file name and file type; for example:

```
readcard junk file c
```

Or, if you want the READ control card to determine the file names and file types, you can enter:

```
readcard * * c
```

10. If you are preparing real or virtual card decks to send to your own or another user's virtual card reader, you may insert READ control cards to designate file names, file types, and optionally, file mode numbers, to be assigned to the file(s).

A READ control card must begin in column 1 and has the format:

```
:READ filename filetype filemode
```

Each field must be separated by at least one blank; the second character of the file mode field, if specified, must be a valid file mode number (0 through 6).

The file mode letter is ignored when this file is read, since the mode letter is determined by specifications on the READCARD command line.

11. To send a real card deck to your own or another user's virtual card reader, punch a CP ID card to precede the deck. The ID card has the keyword ID or USERID in column 1, followed by the user ID you want to receive the file and optionally, spool file class and name designations; for example:
 ID MARY CLASS A NAME LITTLE LAMB
 Each field must be separated from the others by at least one blank.
12. If the reader file being processed contains carriage control characters, the READCARD command returns the records with the carriage control characters stripped off.
13. If you encounter any errors when you read a reader spool file, the file remains in the reader and is not purged by the READCARD command. This occurs regardless of whether you have spooled the reader HOLD or NOHOLD. This protects you from losing reader spool files when an error is encountered. If the file is empty or unwanted, you can purge the file from your reader.

Responses

READCARD issues the following responses, depending on the situation.

1. If the spooled card reader contains no records after the control card:

DMSRDC701I Null file

2. If READCARD * was issued and prompting is *not* in effect, this response indicates that a record beginning with :READ has been found in the spool file and the following file ID is invalid:

DMSRDC702E Missing, invalid, or incomplete fileid in following READ control card:
 :READ...
 Command terminated

3. If READCARD * was issued and a control card was encountered in the input card stream, this response indicates the names assigned to each file:

DMSRDC702I :READ...

4. If READCARD * was issued and the first record in the spool file is not a READ control card, this response is issued when a READ control card in the spool file has been identified and validated, and it is listed at the terminal:

DMSRDC702I READ control card missing. Following assumed:
 :READ READCARD CMSUT1 A1

5. If READCARD * was issued and prompting is in effect, this response indicates that a record beginning with :READ has been found in the spool file and the following file ID is invalid:

DMSRDC702W Missing, invalid, or incomplete fileid in following READ control card:
 :READ...
 Fileid changed to READCARD CMSUT1

6. If the records being read are not 80 bytes long, this message gives the length:

DMSRDC738I Record length is *nnn* bytes

READCARD

7. If you specify the FULLPROMPT or MINPROMPT option one of these prompts will be displayed:

DMSRDC1079R Receive *fn1 ft1 fm1*?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* and replace the existing file
of the same name?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* and replace *fn2 ft2 fm2*?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3*?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and
replace the existing file of the same name?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and replace
fn2 ft2 fm2?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

- The file ID *fn1 ft1 fm1* is the name from the card stream of the spool file.
 - The phrase “and replace the existing file of the same name?” appears when the operation replaces an existing file and the file mode of that file is the same as *fm1*.
 - The phrase “and replace *fn2 ft2 fm2*.” appears when the operation replaces an existing file and the file mode of that file is not *fm1*.
 - The file ID *fn3 ft3 fm3* is the name from the card stream of the spool file that you may specify when the name differs from the name of the incoming file.
8. If you respond with a 3 (or RENAME) to prompt message DMSRDC1079R, the following message appears and you must enter a file ID in the form *fn [ft [fm]]*.

DMSRDC1080R Enter the new name for *fn ft fm*

9. If you respond to prompt message DMSRDC1080R with a file ID that names an existing file, you receive this prompt:

DMSRDC1081R Replace *fn ft fm*?
Reply 0 (NO), 1 (YES), or 2 (QUIT)

10. When READCARD * is entered it will list each READ control card in the spool file and, after it loads an incoming file, it issues one of the following responses. Also, If READCARD *fn ft* is entered it issues one of the following responses.

- If the incoming file (*fn1 ft1 fm1*) does not already exist and is received without being renamed, you receive
fn1 ft1 fm1 created
- If the incoming file (*fn1 ft1 fm1*) is renamed to a file name (*fn2 ft2 fm2*) that does not already exist, you receive
fn2 ft2 fm2 created from *fn1 ft1 fm1*

- If the incoming file (*fn1 ft1 fm1*) is copied to an existing data set that has the same name as the incoming file, you receive
fn1 ft1 fm1 replaced
- If the incoming file (*fn1 ft1 fm1*) is copied to an existing file (*fn2 ft2 fm2*) with a name different from that of the incoming file, you receive
fn2 ft2 fm2 replaced by *fn1 ft1 fm1*
- If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*), but is given a mode (*fm1*) that differs from the mode of the existing file (*fm2*), you receive
fn1 ft1 fm1 replaced *fn2 ft2 fm2*
- If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*), but is given a mode (*fm3*) that differs from the mode of the existing file (*fm2*), you receive
fn3 ft3 fm3 replaced *fn2 ft2 fm2* sent as *fn1 ft1 fm1*

11. When READCARD is issued from the RDRLIST screen, the symbol, /o, must be appended to the READCARD command string. For example,

/o READCARD fn ft fm

See the RDRLIST command for more information about the use of this symbol.

Messages and Return Codes

DMSRDC008E	Device <i>vdev</i> {invalid or nonexistent} is an unsupported device type] [RC = 36]
DMSRDC024E	File <i>fn</i> [<i>ft fm</i>] already exists; specify REPLACE option] [RC = 28]
DMSRDC037E	Filemode <i>mode</i> [(<i>vdev</i>)] is accessed as read/only [RC = 36]
DMSRDC042E	No fileid specified [RC = 24]
DMSRDC054E	Incomplete fileid specified [RC = 24]
DMSRDC062E	Invalid * in fileid [RC = 20]
DMSRDC069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSRDC105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSRDC109S	Virtual storage capacity exceeded [RC = 104]
DMSRDC124S	Error reading card file [RC = 100]
DMSRDC205W	Reader empty or not ready [RC = 8]
DMSRDC257T	Internal system error at address <i>address</i> (offset <i>offset</i>)
DMSRDC639E	Error in <i>routine</i> routine; return code was <i>nnnn</i>
DMSRDC671E	Error loading [file] <i>fn ft fm</i> ; rc = <i>nn</i> from RENAME [RC = 100]
DMSRDC1123E	Unknown response <i>text</i> ignored
DMSRDC1124W	Spool file <i>spoolid</i> has been left in your reader because one or more files were not received [RC = 1]
DMSRDC1138E	Filesharing conflict involving file <i>fn ft fm</i> [RC = 70]
DMSRDC1262S	Error <i>nnn</i> opening file <i>fn ft fm</i> [RC = 31 55 70 99 100]
DMSRDC1262S	Error <i>nnn</i> closing file <i>fn ft fm</i> [RC = 31 100]
DMSRDC1285S	Default option <i>text</i> is invalid [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

RECEIVE

Use the RECEIVE command to read onto a disk or directory one of the files or notes that is in your virtual reader. In most cases these files were sent to you by other computer users, on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network.

Format

RECEIVE	<p>[<i>spoolid</i> [<i>fn</i> [<i>ft</i> [<i>fm</i>]]]] [(options... [])]</p> <p>Options:</p> <p>[<u>NOTebook</u> <i>fn</i>] [<u>Log</u>] [Purge] [<u>Fullprompt</u>] [<u>Replace</u>] [NOTebook *] [<u>NOLog</u>] [<u>Minprompt</u>] [<u>NOReplace</u>] [<u>NOPrompt</u>]</p> <p>[<u>Olddate</u>] [<u>STack</u>] [<u>NEwdate</u>]</p>
----------------	---

Operands

spoolid

specifies which file in the virtual reader is to be received. The default is '=' or 'next' which means the 'next' file in the reader is received.

The 'next' file is the one for which the RDR command returns information. Which file this is depends on the class of the reader, the class of the files in the reader, and whether or not they are held.

To give the file a new file identifier, you must specify the spoolid, '=', or 'next'.

fn

is the file name the file is to be given. The default is =, which means the file's present name is used.

ft

is the file type the file is to be given. The default is =, which means the file's present type is used.

fm

is the file mode the file is to be given. If not specified, the default is A.

If the file being received is a note (prepared by the NOTE command), or if the PURGE option is specified, the operands *fn ft fm* are ignored. If the file being received is an acknowledgment, all parameters and all options (except the spoolid and the PURGE option) are ignored. For more information about acknowledgments see the usage note, Acknowledgments.

Options

NOTebook *fn*

causes the file to be saved as a note in a file named *fn* NOTEBOOK. You can use this option if you want the note(s) from this person to be kept in a separate file. If you do not specify a notebook file name here, a file name is first searched for in the sender's entry in your *userid* NAMES file and then in a file set up by the DEFAULTS command. If neither contains a notebook file name, the note is saved in the default notebook file, ALL NOTEBOOK. A note is saved by appending it to the NOTEBOOK file, with a line of 73 equal signs (=) separating each note.

If the file is not a note (prepared by the NOTE command), this option is ignored.

See the NAMEFIND or NAMES command description for more information on the relationship between a *userid* NAMES file and the NOTEBOOK file.

NOTebook *

specifies that note is saved in a file named *name* NOTEBOOK, where *name* is the value of the Notebook tag in the sender's entry in your *userid* NAMES file, or the sender's nickname, or the sender's user ID (whichever is located first).

If the file is not a note (prepared by the NOTE command), this option is ignored.

Log

specifies that the recipients, date, and time of this file transmission are logged in a file called *userid* NETLOG. This log is updated when acknowledgments of sent files are received (if they were requested). You must have a read/write disk or directory accessed to use this option.

NOLog

specifies that this file transmission is not to be logged.

Purge

specifies that this file is to be purged and not read onto a disk or directory.

Fullprompt

specifies that a prompt is issued for each file without regard to the format of the spool file being processed.

If the file being received is a NOTE file, FULLPROMPT is ignored.

Minprompt

specifies that a prompt is issued when the name of the first (or only) file differs from the name of the spool file; the prompt for the first file is suppressed when it has the same name as the spool file. A prompt is always issued for the second and subsequent files. This prompt is issued only for files in DISK DUMP and NETDATA format. MINPROMPT is the default.

If the file being received is a NOTE file, MINPROMPT is ignored.

NOPrompt

specifies that a prompt is not issued to you as a file is received.

Replace

specifies that if a file of the same file name and file type exists on the disk or directory onto which the incoming file is to be loaded, it is to be replaced with this one.

NOReplace

specifies that a file is not received that would overlay an existing file on the receiving disk or directory. **NOREPLACE** is the default.

Olddate

means that when a file that was sent in **DISK DUMP** format is received, it is written to a disk or directory with its original date and time (that is, the date and time it was created or last updated by the sender), not the date and time you received it. When a file that was sent in **NETDATA** format is received, it is written to a disk or directory with its original date and time unless it was sent from a different time zone, in which case the date and time are changed to reflect **GMT** (Greenwich Mean Time). See the usage note, *Receiving NETDATA Files Using the Olddate Option*. For more information on **NETDATA** format, see the *VM/SP CMS Diagnosis Reference*

NEwdate

means to re-date the file to the current date and time it is received.

STack

specifies that the message returned when **RECEIVE** completes successfully should be stacked (**LIFO**). If this option is not specified, the messages from **RECEIVE** are displayed at the terminal.

Usage Notes**1. Tailoring the RECEIVE Command Options**

You can use the **DEFAULTS** command to set up options and/or override command defaults for **RECEIVE**. However, the options you specify in the command line when entering the **RECEIVE** command override those specified in the **DEFAULTS** command. This allows you to customize the defaults of the **RECEIVE** command, yet override them when you desire. Refer to the **DEFAULTS** command description for more information.

2. Why Should I Use Receive?

You should use **RECEIVE** instead of **READCARD** or **DISK** for general purpose use, because **RECEIVE** calls either **READCARD** or **DISK**, whichever is appropriate. It also handles notes, acknowledgments, etc. In fact, **RECEIVE** handles most of the various formats of files that can appear in your virtual reader. **RECEIVE** is the only way to read a file that was sent using the **SENDFILE** command issued with the **NEW** option.

RECEIVE is particularly useful within the **RDRLIST** command environment, where it is assigned to the **PF9** key.

You may send multiple files by continuous spooling (using **CP SPOOL PUNCH CONT**) or by a series of **DISK DUMP** commands but those methods are discouraged. As a sender, you are encouraged to do the following:

- Always use **SENDFILE**, which resets any continuous spooling options in effect.
- Do not spool the punch continuous.

Similarly, if the punch is spooled continuous and **PUNCH** is used to send multiple files, the file is read in as one file with **:READ** cards imbedded. In this case, although no files are overlaid, the recipient must divide the file into individual files. This problem can also be avoided by using **SENDFILE** or by not spooling the punch continuous.

3. Receiving Multiple Files

You cannot receive multiple files as one file by spooling your reader continuous (CONT). The RECEIVE command resets the continuous spooling option and spools your reader NOCONT.

4. Acknowledgments

Acknowledgments can be sent to users on different computers connected by the RSCS network so that they can be sure that a file they sent was received.

The sender can specify on the SENDFILE or NOTE command that an acknowledgment be returned to him when a file is received. The SENDFILE command must be issued with the NEW option (the default) in order to request an acknowledgment; otherwise, the request is ignored. Even if a recipient discards a file (using the DISCARD command), an acknowledgment is returned to the sender. This is possible because DISCARD is equivalent to a RECEIVE issued with the PURGE option. (For more information on DISCARD, see the RDRLIST command.) The acknowledgment indicates whether the file was received (written to a disk or directory) or discarded (purged).

When you RECEIVE an acknowledgment that appears in your reader, all parameters and all options (except the spoolid and the PURGE option) are ignored. The acknowledgment is used to make an entry in your *userid* NETLOG file. This entry confirms that the file you sent was received (or discarded). The format of entries in the *userid* NETLOG file is shown in the Examples section, below.

5. Responding to Prompting Messages

If you specify the FULLPROMPT or MINPROMPT option the valid responses include:

- One of the digits specified in the prompt
- One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	If this file is one of a set of files that constitutes a single spool file, the file is not received and prompting continues for the next file, if there is one. If this is the last file of a set of files or if this is the only file in the spool file, the command is ended.
1 or Yes	Receives the file under the name <i>fn1 ft1 fm1</i> (or <i>fn3 ft3 fm3</i>).
2 or Quit	Ends the command.
3 or Rename	Requests prompt message DMSWRC1080R so the incoming file can be received using a different name.

6. If you receive prompt message DMSWRC1081R the valid responses include:

- One of the digits specified in the prompt
- One of the parenthetical words that follows a digit or any initial truncation of the word.

The meanings of these responses are:

Response	Description
0 or No	Does not receive the file under the name <i>fn ft fm</i> and repeats the original prompt message DMSWRC1080R. This allows you to specify a different name for the incoming file.
1 or Yes	Receives the file under the name <i>fn ft fm</i> .
2 or Quit	Ends the command.

7. Which prompt is most useful for you?

- If you do not issue either a QUERY RDR command or a RDRLIST command specify FULLPROMPT.
- If you do issue a QUERY RDR command before issuing the RECEIVE command or if you issue RECEIVE from a RDRLIST screen specify MINPROMPT.
- If you issue the RECEIVE command in a controlled environment where the identity of all incoming files are known, specify NOPROMPT.

8. Special NETDATA Files from MVS with TSO Extensions (PP)

The MVS with TSO Extensions Program Product can send an empty file, in which case RECEIVE will give you an error message indicating that no file was created on a disk or directory. It can also send, as a unique case of multiple files in one transmission, one note and a data file together. The note will be the first file in the transmission and the data file will be second. RECEIVE will add the note to the appropriate notebook, receive the data file, and give informative messages for each action. This is the only form of multiple NETDATA files supported by the RECEIVE command.

Note: RECEIVE will not handle partitioned data sets or data sets that have been encrypted by Access Method Services. These files will not be received and remain in the reader. Multiple NETDATA transmissions that do not have a note as the first file, result in the first file being received and the other file(s) ignored. The entire spool file is left in the user's reader.

9. How does RECEIVE determine the file ID?

If you do not specify a *fn ft fm*, then RECEIVE determines the file ID according to the method that was used to send the file.

File Format	How the FILE ID is Determined
NETDATA files	Uses the file ID that was specified on the SENDFILE command.
DISK DUMP files	Uses the file ID that was specified on the DISK DUMP command.
CONSOLE, PUNCH, and PRINTER files	Uses the <i>fn</i> and <i>ft</i> from the QUERY RDR ALL response.

If the file is not NETDATA format and the file type is NOTE or MAIL, the file is considered to be a note and is put in the notebook file.

If you are not sure of the method used to send a file that is in your virtual reader, then use the RDR command.

10. The RECEIVE command does not handle MONITOR files or files with a SPECIAL status of YES. (The SPECIAL status indicates whether or not the file

contains records with X'5A' carriage control characters. See the CP QUERY command to determine SPECIAL status of a file.)

11. Receiving NETDATA Files Using the Olldate Option

If a file is sent in NETDATA format using SENDFILE from one location to another in a different time zone, and it is received using the OLDDATE option, the date and time of the file reflects when it was last modified relative to GMT (Greenwich Mean Time).

For example, suppose a file was last modified on 01/01/86 14:00 in a time zone that is 8 hours west of GMT, and it is sent using SENDFILE to a time zone 5 hours west of GMT. When the file is received, the time and date is changed to 01/01/86 17:00.

12. If you want to issue RECEIVE from an exec program, you should precede it with the EXEC command; that is, specify

```
exec receive
```

13. If you encounter any errors when you receive a reader spool file, the file remains in the reader and is not purged by the RECEIVE command. This occurs regardless of whether you have spooled the reader HOLD or NOHOLD. This protects you from losing reader spool files when an error is encountered. If the file is empty or unwanted, you can purge the file from your reader.

Examples

Format of the userid NETLOG File:

The format of entries in the *userid* NETLOG file maintained by SENDFILE and RECEIVE is shown below. If both the ACK and LOG options of SENDFILE or NOTE are specified, a sent to record is placed in the NETLOG file. When an acknowledgment is received, it is also placed in this file.

```
File SMALL DATA A sent to OHARA at NODE01 on 10/14/80 11:30:25
File SMALL DATA A rcv from OHARA at NODE01 on 10/14/80 11:30:47
sent as SMALL DATA A
Ackn 10/14/80 11:30:47 rcv by OHARA at NODE01 on 10/14/80 11:30:25
```

In this example, the user sent himself a file (SMALL DATA) using SENDFILE with the LOG and ACK options specified. The first line in the NETLOG file was placed in the file by the SENDFILE command.

He then used RECEIVE (with the LOG option) to read the file onto his disk or directory accessed as A. The second line was added when the file was received. (In this case the sender was the receiver.) The rcv in this line means received. If a file is discarded (using DISCARD), the line contains disc instead of rcv. (The file can be received with a different file ID than it was sent as.)

Last, he received an acknowledgment. It indicates whether the recipient received (rcv) or discarded (disc) the file.

Responses

1. If you specify the FULLPROMPT or MINPROMPT option one of these prompts is displayed:

DMSWRC1079R Receive *fn1 ft1 fm1*?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

 Receive *fn1 ft1 fm1* and replace the existing file
 of the same name?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

 Receive *fn1 ft1 fm1* and replace *fn2 ft2 fm2*?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

 Receive *fn1 ft1 fm1* as *fn3 ft3 fm3*?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

 Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and replace
 the existing file of the same name?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

 Receive *fn1 ft1 fm1* as *fn3 ft3 fm3* and replace
 fn2 ft2 fm2?
 Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

- The file ID *fn1 ft1 fm1* is the name from the card stream of the spool file.
 - The phrase “and replace the existing file of the same name?” appears when the operation replaces an existing file and the file mode of that file is the same as *fm1*.
 - The phrase “and replace *fn2 ft2 fm2*.” appears when the operation replaces an existing file and the file mode of that file is not *fm1*.
 - The file ID *fn3 ft3 fm3* is the name from the card stream of the spool file that you may specify when the name differs from the name of the incoming file.
2. If you respond with a 3 (or RENAME) to prompt message DMSWRC1079R, this message appears and you must enter a file ID of the form *fn [ft [fm]]*.

DMSWRC1080R Enter the new name for *fn ft fm*

3. If you respond to prompt message DMSWRC1080R with a file ID that names an existing file, you receive this prompt:

DMSWRC1081R Replace *fn ft fm*?
 Reply 0 (NO), 1 (YES), or 2 (QUIT)

4. Spool files in the DISK DUMP or PUNCH format will issue the following response.

File *fn1 ft1 fm1* received from *userid* at *node* sent as *fn2 ft2 fm2*

5. Spool files in the NETDATA format issue any one of the following responses.

- If the incoming file (*fn1 ft1 fm1*) does not already exist:

File *fn2 ft2 fm2* created from *fn1 ft1 fm1* received from
userid at *node*

- If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*):

File *fn2 ft2 fm2* replaced by *fn1 ft1 fm1* received from *userid* at *node*

- If the incoming file (*fn1 ft1 fm1*) replaces an existing file (*fn2 ft2 fm2*), but is given a mode (*fm3*) that differs from the mode of the existing file (*fm2*):

File *fn3 ft3 fm3* replaced *fn2 ft2 fm2* with *fn1 ft1 fm1* received from *userid* at *node*

6. RECEIVE issues the following responses for spool files in the DISK DUMP format if you specify the operands *spoolid fn ft fm* and if the incoming file is given a different name or placed on a disk or directory other than your file mode A.

- If the first (or only) file in the spool file has the same file name and file type as an existing file on your disk or directory accessed as A, you will get these responses:

fn1 ft1 fm1 created
fn1 ft1 fm1 saved in a temporary file

- If an error does not occur you receive this response so that you are aware that your original file has not been destroyed:

fn1 ft1 fm1 copied to *fn2 ft2 fm2* and original *fn1 ft1 fm1* restored

- If an error does occur you receive this response and RECEIVE resumes processing.

original *fn1 ft1 fm1* restored

- If the first (or only) file in the spool file has a file name and file type that is not the name of an existing file on your disk or directory accessed as A, and the incoming file is to be placed on a disk or directory other than your file mode A, you will get these responses.

fn1 ft1 fm1 created
fn1 ft1 fm1 copied to *fn2 ft2 fm2* and *fn1 ft1 fm1* then erased

7. Other responses include:

File *spfn spft* has been discarded

Note *spfn spft* has been discarded

Note *spfn spft* added to *fn NOTEBOOK fm*

Ackn *date time* added to *userid NETLOG*

Ackn *date time* has been discarded

Messages and Return Codes

DMSWRC006E	No read/write filemode accessed [RC = 36]
DMSWRC024E	File <i>fn [ft fm]</i> already exists[; specify REPLACE option] [RC = 28]
DMSWRC037E	Filemode <i>mode</i> is accessed as read/only[; A must be R/W for DISK LOAD] [RC = 36]
DMSWRC062E	Invalid {character [<i>char</i>]*} in [output] fileid [<i>fn ft [fm]</i>] [RC = 20]
DMSWRC069E	Filemode <i>mode</i> not accessed [RC = 36]
DSMWRC630S	Error accessing spool file [RC = 36]
DMSWRC643E	No class <i>class</i> files in your reader [RC = 28]
DMSWRC644E	All reader files are in HOLD status or not class <i>class</i> [RC = 28]
DMSWRC655E	Spoolid <i>nmmn</i> does not exist [RC = 28]

RECEIVE

- DMSWRC671E Error receiving file *fn ft fm*; rc=*nn* from *command* [RC = 100]
- DMSWRC672E Virtual reader invalid or not defined [RC = 36]
- DMSWRC674E Reader is not ready [RC = 36]
- DMSWRC681E This is an unnamed file; specify filename and filetype [RC = 88]
- DMSWRC682E Error copying file *fn ft A* to *{fn ft fm|mode}*; rc=*nn* from COPYFILE [RC = 100]
- DMSWRC687E This is a {SYSTEM{HELD|DUMP}file|file with a special format} and cannot be received [RC = 1]
- DMSWRC1123E Unknown response *text* ignored
- DMSWRC1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC = 1]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

RELEASE

Use the **RELEASE** command to free an accessed disk or Shared File System (SFS) directory that was previously accessed with the **ACCESS** command.

Format

RELease	$\left. \begin{array}{l} vdev \\ dirid \\ fm \end{array} \right\}$	[(DET[])]
----------------	--	------------

Note: The **DET** option is ignored when you are releasing a directory.

Operands*vdev*

is the virtual device address of the disk that is to be released.

Valid *vdev* addresses are:

- 0001 through FFFF for a 370/XA mode virtual machine
- 001 through 5FF for a VM/SP virtual machine in basic control mode
- 001 through FFF for a System/370 mode virtual machine or VM/SP virtual machine in extended control mode.

On VM/SP and System/370 mode virtual machines you can supply a leading zero. For example,

```
access 0191 a
```

Note: In the preceding description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. Valid addresses in both environments are described to help you plan and develop applications that will run in both environments.

dirid

is the name of the directory to be released. See "Naming Shared File System (SFS) Directories" on page 4 for a complete description of *dirid*.

fm

is the file mode letter for which the disk or directory is currently accessed.

Options**DET**

specifies that the disk is to be detached from your virtual machine configuration; CMS calls the CP command **DETACH**. The **DET** option is ignored when releasing an SFS directory.

RELEASE

Usage Notes

1. If a disk or directory is accessed at more than one file mode, the `RELEASE vdev` or `RELEASE dirid` command releases all modes. If you specify the file mode of an already active disk or directory, that disk or directory is released.
2. You cannot release the system disk (disk at file mode S).
3. A list of the files on an accessed disk or directory is maintained in storage. When a disk is released, this list of files is freed from storage and that storage becomes available for other CMS commands and programs. When a directory is released, the list is freed from storage only when and if other CMS commands or programs need the storage.

When you release a read/write CMS disk either with the `RELEASE` command or implicitly with the `FORMAT` command or `ACCESS` commands, the list of files is sorted and rewritten on disk; user(s) who may subsequently access the same disk may have a resultant favorable decrease in file search time.
4. When a disk or directory is released, any read-only extensions it has are not released. The extensions may be referred to by their own file mode. If a disk or directory is then accessed with the same file mode, the original read-only extensions remain extensions to the new disk or directory at that file mode.
5. In CMS/DOS, when you release a disk, any system or programmer logical unit assignments made for the disk are unassigned.
6. The `RELEASE` command rewrites the file directory on any CMS disk accessed in R/W mode whether or not the disk was altered. The exception to this is when the `ACCESS` command has been issued with the `ERASE` option for a disk. If no files have been written to the disk since issuing `ACCESS` with the `ERASE` option, `RELEASE` will not rewrite the directory (see usage notes for the `ACCESS` command).

Examples

If you want to release and detach the 498 disk that is accessed as your file mode B, then issue:

```
release 498 (det
```

or

```
release b (det
```

If you want to release the directory `.PROJECT1` that is accessed as your file mode C, enter:

```
release .project1
```

or

```
release c
```

Responses

```
DASD vdev DETACHED
```

This is a CP message that is issued when you use the `DET` option. It indicates that the disk has been detached.

Messages and Return Codes

DMSARE028E No device specified [RC=24]
DMSARE048E Invalid mode *mode* [RC=24]
DMSARE069E Filemode *mode* not accessed [RC=36]
DMSARE069E Disk *vdev* not accessed [RC=36]
DMSARE069E Directory *dirname* not accessed [RC=36]
DMSARE070E Invalid parameter *parameter* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

RELOCATE

RELOCATE

Use the RELOCATE command to:

- move one or more of your files from one of your Shared File System (SFS) directories to another of your directories
- move one of your directory structures to another directory that you own.

Format

RELOCate	$\left[\begin{array}{l} fn\ ft \\ * \ * \end{array} \right] \text{ dirid1 TO dirid2 } [(\text{options...} [])]$ <p>Options:</p> $\left[\begin{array}{l} \text{TYPE} \\ \text{NOType} \\ \text{STACK} \left[\begin{array}{l} \text{FIFO} \\ \text{LIFO} \end{array} \right] \\ \text{FIFO} \\ \text{LIFO} \end{array} \right]$
-----------------	--

Operands

fn ft

is the name and type of the file to be moved.

You can use special characters (* or %) to designate a group of files. See “Pattern Matching” on page 8 for information on these special characters.

Not specifying *fn ft* indicates that you are relocating a directory or directory structure.

dirid1

is either:

- the name of the directory that contains the file(s) to be moved, if you specify *fn ft*
- or
- the name of the parent directory (top node) of the directory structure that is to be moved, if you do not specify *fn ft*.

Your top directory cannot be moved. You must be the owner of *dirid1*.

When you relocate a directory structure, all subdirectories and files move with the directory structure.

For a complete description of *dirid*, see the “Naming Shared File System (SFS) Directories” on page 4.

dirid2

is either:

- the name of the directory that will contain the file(s) being moved, if you specify *fn ft*

or

- the name of the new parent directory, if you do not specify *fn ft*.

You must be the owner of *dirid2*. The target directory, *dirid2*, cannot be the same as *dirid1*. When relocating a directory, the new parent directory (*dirid2*) cannot be a parent or a subdirectory of *dirid1*. Both directories must be in the same file pool.

Options

Type

displays information at the terminal.

NOType

suppresses the display of information at the terminal. NOTYPE is the default.

STACK **[FIFO]**

STACK LIFO

places the information in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

Usage Notes

1. When you move a base file or a directory structure, all authorities and aliases remain the same.
2. When you move an alias, the alias remains in effect.
3. Erased and revoked aliases can be relocated.
4. If you or other users have a directory accessed that gets relocated, the access remains in effect.
5. Any file(s) that you are moving must be closed. If you are relocating a directory structure, all files in the directory structure must be closed. If you use special characters to specify a set of files, the directory (*dirid1*) cannot be open.
6. If a file, or the directory containing the file, is locked, you cannot relocate it unless you have an UPDATE or EXCLUSIVE lock on it. You cannot move a file that is locked by another user; and you cannot move a file that resides in a directory that is locked by another user.
7. If a directory, or any of its subdirectories, are locked you cannot relocate it unless you have an UPDATE or EXCLUSIVE lock on them. You cannot move a directory that is locked by another user, or has any of its subdirectories locked by another user.
8. When moving a directory structure, the resulting directory structure cannot be more than eight layers deep (nine layers if you count the top directory). For example, assuming *.onelayer* is a directory with no subdirectories, relocate *.onelayer* to *.n1.n2.n3.n4.n5.n6.n7*

RELOCATE

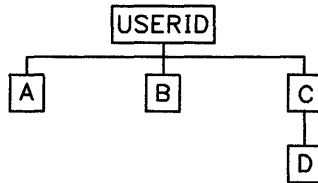
is valid and would result in a directory of eight qualifiers
(.n1.n2.n3.n4.n5.n6.n7.onelayer). But the following command, assuming
.twolayer has at least one subdirectory (.twolayer.twolayersubdir),
relocate .twolayer to .n1.n2.n3.n4.n5.n6.n7

would result in directory of nine qualifiers
(.n1.n2.n3.n4.n5.n6.n7.twolayer.twolayersubdir) and is not valid.

9. If the RELOCATE command is issued from an exec on a work unit that is active, the command will fail.
10. You can invoke the RELOCATE command from the command line, from an exec, or as a function from a program. No error messages are issued if RELOCATE is invoked:
 - As a function from a program
 - From a CMS exec file that has the &CONTROL NOMSG option in effect
 - From an EXEC2 exec where CMDCALL is not in effect
 - From a System Product Interpreter exec with ADDRESS COMMAND in effect

Example

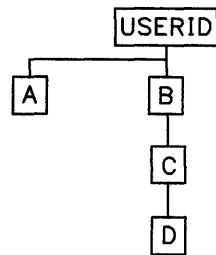
If you started with the following directory structure:



and issued,

```
relocate .c to .b
```

you would get this directory structure:



Messages and Return Codes

DMSJED069E	Filemode <i>fm</i> not accessed [RC = 36]
DMSJED109S	Virtual storage capacity exceeded [RC = 104]
DMSJED1187E	Too many subdirectory levels in <i>dirid</i> [RC = 24]
DMSJED1188E	Filemode <i>fm</i> is not associated with a directory [RC = 74]
DMSJED1189E	Filemode <i>fm</i> is associated with a top directory [RC = 24]
DMSJED1223E	There is no default file pool currently defined [RC = 40]
DMSJRL002E	File <i>fn ft fm dirname</i> not found [RC = 28]

DMSJRL019E Identical fileids [RC=24]
 DMSJRL1131E Directory *dirname* already exists [RC=28]
 DMSJRL1132E Invalid number of operands [RC=24]
 DMSJRL1160E Directory *dirname* already open. [RC=70]
 DMSJRL1163E The RELOCATE command failed for *fn ft fm|dirname* [RC=00]
 DMSJRL1184E File *fn ft* or directory *dirname* not found or you are not
 authorized to use RELOCATE on one of these directories
 [RC=28]
 DMSJRL1187E Too many subdirectory levels in *dirid* [RC=24]
 DMSJRL1207E You cannot relocate a top directory [RC=88]
 DMSJRL1208E Directory cannot be relocated within itself [RC=24]
 DMSJRL1210E Directory *dirname* or directory *dirname* not found or you are not
 authorized to use RELOCATE on one of these directories.
 [RC=28]
 DMSJRL1241E Directories specified are in different file pools [RC=88]
 DMSJRL1251E Directories are from different directory structures [RC=88]
 DMSJRL1290E File *fn ft fm|dirname* not relocated; source and target directories
 are the same [RC=24]
 DMSJRL1291E There are no unused work units available. [RC=88]
 DMSOUT065E *option* option specified twice [RC=24]
 DMSOUT066E *option1* and *option2* are conflicting options [RC=24]
 DMSOUT105S Error *nn* writing file to XEDIT [RC=100]
 DMSOUT1201E STACK option cannot follow FIFO or LIFO [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in parsing a command	397

RENAME

Use the RENAME command to:

- change the file ID of one or more files on a read/write minidisk or Shared File System (SFS) directory. The file can be an alias.
- change the name of an SFS directory that you own.

Format

Rename	$\left\{ \begin{array}{ll} \textit{fileid1} & \textit{fileid2} \\ \textit{dirid1} & \textit{dirid2} \end{array} \right\} \quad [(\textit{options...} [])]$										
	<p>Options:</p> <table border="0" style="margin-left: 40px;"> <tr> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 5px;"> Type NOType STACK </td> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 5px;"> <table border="0" style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">FIFO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">LIFO</td> </tr> </table> </td> <td style="border-left: 1px solid black; padding: 0 5px;"> <table border="0" style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">UPdirt</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">NOUPdirt</td> </tr> </table> </td> </tr> <tr> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 5px;"> FIFO LIFO </td> <td></td> <td></td> </tr> </table>	Type NOType STACK	<table border="0" style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">FIFO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">LIFO</td> </tr> </table>	FIFO	LIFO	<table border="0" style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">UPdirt</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">NOUPdirt</td> </tr> </table>	UPdirt	NOUPdirt	FIFO LIFO		
Type NOType STACK	<table border="0" style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">FIFO</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">LIFO</td> </tr> </table>	FIFO	LIFO	<table border="0" style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">UPdirt</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">NOUPdirt</td> </tr> </table>	UPdirt	NOUPdirt					
FIFO											
LIFO											
UPdirt											
NOUPdirt											
FIFO LIFO											

Operands

fileid1

is the file identifier of the file you want to rename. For a file on a minidisk, the file identifier consists of a file name, file type, and file mode. For a file in an SFS directory, the file identifier consists of a file name, file type, and a *dirid*. See “Naming Shared File System (SFS) Directories” on page 4 for a complete description of *dirid*.

You can use an asterisk (*) for any part of the file identifier to indicate that any file that satisfies the other qualifications is renamed. Subdirectories that match the specified pattern are ignored and are not renamed.

fileid2

is the new file identifier of the file. For a file on a minidisk, the file identifier consists of a file name, file type, and file mode. For a file in an SFS directory, the file identifier consists of a file name, file type, and a *dirid*.

You can use an equal sign (=) for any part of the file identifier to indicate that the corresponding file identifier is unchanged. The file mode or *dirid* can also be specified as an asterisk (*), indicating that the corresponding file mode or *dirid* is not changed.

dirid1

is the directory identifier of the SFS directory you want to rename. See “Naming Shared File System (SFS) Directories” on page 4 for a complete description of *dirid*.

dirid2

is the new directory name. The file mode cannot be expressed as *dirid2* when you rename a directory.

Options

Type

displays, at the terminal, the new identifiers of all the files that are renamed. The file identifiers are displayed only when an asterisk (*) is specified for one or more of the file identifiers (*fn*, *ft*, *fm*, or *dirid*) in *fileid1*.

NOType

suppresses the display at the terminal of the new file identifiers of the renamed files. NOTYPE is the default.

STACK [FIFO]

STACK LIFO

places the information in the console stack rather than displaying it at the terminal. The new file identifiers are stacked only when an asterisk (*) is specified for one or more of the file identifiers (*fn*, *ft*, *fm*, or *dirid*) in *fileid1*. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

UPdir

updates the list of files upon completion of this command.

UPDIRT is not valid when you are renaming a file in a directory or the directory itself.

NOUPdir

suppresses the updating of the master file directory when a file on a minidisk is renamed. (See Usage Note 13.)

NOUPDIRT is not valid when you are renaming a file in a directory or the directory itself.

Usage Notes

1. When you code an asterisk (*) in any portion of the input file ID, any or all of the files that satisfy the other qualifiers may be renamed, depending upon how you specify the output file ID. For example:

```
rename * assemble a test file a
```

results in the first ASSEMBLE file found on the minidisk or directory accessed as A being renamed to TEST FILE. If more than one ASSEMBLE file exists, error messages are issued to indicate that they cannot be renamed.

If you code an equal sign (=) in an output file ID in a position corresponding to an asterisk in an input file ID, all files that satisfy the condition are renamed. For example:

```
rename * assemble a = oldasm =
```

renames all files with a file type of ASSEMBLE to files with a file type of OLDASM. Current file names are retained.

2. You cannot use the RENAME command to move a file from one minidisk or directory to another. For files on minidisks, you must use the COPYFILE command if you want to copy a file to another minidisk. For files in directories,

RENAME

you can use the RELOCATE command to move a file to another directory or the COPYFILE command to copy the file to another directory.

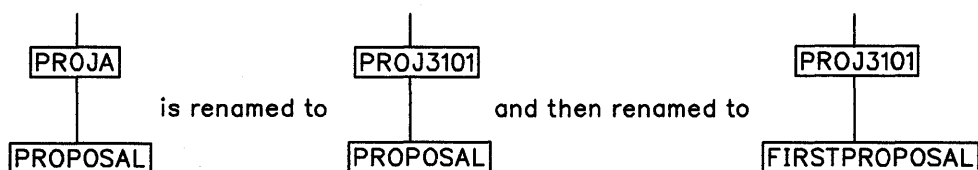
Similarly, you cannot use the RENAME command to move an SFS directory to another parent directory. You must use the RELOCATE command if you want to move a directory structure.

3. You can use the RENAME command file to modify file mode numbers for base files. You cannot use the RENAME command to modify the file mode number for an alias. Aliases are automatically updated with the same file mode number as the base file. For example,

```
rename * module a1 = = a2
```

changes the file mode number on all MODULE files that have a mode number of 1 to a mode number of 2.
4. When you rename an alias, only the alias is renamed; the base file name stays the same.
5. You can rename erased or revoked aliases or you can rename existing files to the name of an erased or revoked alias.
6. You can invoke the RENAME command from the terminal, from an exec file, or as a function from a program. If RENAME is invoked as a function or from an exec file that has the &CONTROL NOMSG option in effect, the message DMSRNM002E (File *fn ft fm* not found) is not issued.
7. When you rename a directory, all authorizations for you and other users remain in effect. QUERY SEARCH or QUERY ACCESSED will display the new directory name.
8. When you rename a file in a directory, the file must be closed, and the directory may be open or closed. If special characters are specified to rename a set of files, the directory must be closed to rename the files that match the pattern.
9. When you rename a directory, all files in the directory must be closed and the directory and all the subdirectories must either be closed or open with intent of FILE.
10. If a file or the directory containing the file is locked you cannot rename the file unless the lock is an UPDATE or EXCLUSIVE lock that you hold.
11. If a directory is locked, or contains locked files or subdirectories, the directory cannot be renamed unless the lock is an UPDATE or EXCLUSIVE lock that you hold. These same rules apply for renaming a subdirectory in a locked directory.
12. Directories can only be renamed one level at a time. For example, changing .PROJA.PROPOSAL to .PROJ3101.FIRSTPROPOSAL would require two commands:

```
rename .proj3101 .proj3101
rename .proj3101.proposal .proj3101.firstproposal
```



As illustrated by the first rename above, changing the name of a directory changes the names of all of its subdirectories.

13. Normally, the list of files in memory for a CMS minidisk is updated whenever you issue a command that affects files on the minidisk. If you use the NOUPDIRT option of the RENAME command, this list is not updated until you issue a command that writes, updates, or deletes any file on the minidisk, or until you explicitly release the minidisk (with the RELEASE command).

Note: The NOUPDIRT option is only valid for files on a minidisk; it is ignored when you rename files in a directory or when you rename a directory.

14. You can only rename directories that you own. You can rename another user's file if you have write authority to the file and write authority to the directory that contains the file. You can also rename another user's alias, if you have read authority to the base file and write authority to the directory containing the alias.
15. When renaming a file in another user's directory, you cannot use the *fm* form of *dirid*.
16. You can issue RENAME from the command (Cmd) column on any of the FILELIST screens. For example, to rename a file that is located in another user's directory name that is displayed on the screen, enter:

```
rename /ntd newname = =
```

This indicates that you wish to rename the *fn ft* directory to a new file name and keep the same file type and directory name as displayed. You must specify either the *d* or the directory name since specifying *fm* is not valid.

To rename a file in your own directory, enter:

```
rename / newname = =
```

17. When renaming a directory, if the RENAME command is issued from an exec on a work unit that has not been committed, the command will fail.

Responses

```
newfn newft newfm
```

The new file name, file type, and file mode of each file altered is displayed when the TYPE option is specified and an asterisk was specified for at least one of the file identifiers (fn, ft or fm) of the input file ID.

Messages and Return Codes

- DMSRND002E File *fn ft fm|dirname* not found [RC = 28]
- DMSRND019E Identical fileids [RC = 24]
- DMSRND051E Invalid mode change [RC = 24]
- DMSRND1131E Directory *dirname* already exists [RC = 28]
- DMSRND1160E Directory *dirname* already open. [RC = 70]
- DMSRND1184E File *fn ft fm|dirname* not found or you are not authorized for it [RC = 28]
- DMSRND1184E Directory *dirname* not found or you are not authorized for it [RC = 28]
- DMSRND1199E You cannot rename a top directory [RC = 24]
- DMSRND1226E Invalid subdirectory name change. Only the last qualifier of the specified subdirectory can be renamed. [RC = 24]
- DMSRND1241E Directories specified are in different file pools [RC = 88]
- DMSRND1257E The RENAME command is invalid on a file in a directory that you do not own [RC = 76]

RENAME

- DMSRND1257E The RENAME command is invalid on a directory that you do not own [RC=28]
- DMSRND1308E The filemode number of an alias must be the same as the filemode number of the base file [RC=24]
- DMSRND1309I Command completed successfully, but the filemode number of the alias is the same as the filemode number of the base file [RC=0]
- DMSRNM030E File *fn ft fm* already active [RC=28]
- DMSRNM037E Filemode *mode* is accessed as read/only [RC=36]
- DMSRNM048E Invalid filemode *mode* [RC=24]
- DMSRNM054E Incomplete fileid specified [RC=24]
- DMSRNM069E Filemode *mode* not accessed [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811
Errors in the Shared File System	813

RESERVE

Use the RESERVE command to allocate all available blocks of a 512-, 1K-, 2K-, or 4K-byte block formatted minidisk to a unique CMS file.

Format

RESERVE	<i>fn ft fm</i>
----------------	-----------------

Operands

fn
is the file name of the file to which you are allocating the available blocks.

ft
is the file type of the file to which you are allocating the available blocks.

fm
is the file mode of the file to which you are allocating the available blocks. If you do not specify a file mode number (0-6), then the file mode number defaults to 6.

Format of the File

When the RESERVE command completes, the defined file has the following format:

- file name, file type, and file mode letter as defined by the user.
- file mode number 6 (indicating “update in place”) if not specified on the RESERVE command.
- logical record length (lrecl) equal to the CMS minidisk block size.
- fixed (F) record format.
- the number of records is the total number of blocks available on the disk minus the number of blocks used by CMS. You can use the DISKID function to get this number. This CMS overhead varies with the size of the minidisk. The data blocks physically follow the blocks used by CMS.

The file that is created can be read or written via the DASD Block I/O System Service or the CMS file system. Because a CMS file structure has been created on the disk, the file may be accessed using the CMS file system.

Organization of the Mini-disk

When the RESERVE command completes, the physical organization of the minidisk on a CKD device is:

RESERVE

Start Block	No. of Blocks	Description
1	2	IPL record
3	1	Volume label
4	2	CMS file directory
6	*	Pointer blocks for allocation map
next	*	CMS allocation map
next	*	Duplicate for CMS allocation map backup
next	*	File pointer blocks
next	*	File data blocks

Example

Suppose you have a 3330 device with one cylinder formatted with 1024-byte block size. There will be 209 blocks available. After you issue the RESERVE command, the file created will have the following format:

```
Blocks used by CMS file system | data blocks
1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | ... | 207 | 208 | 209
```

where:

Physical block Number	Description
1 and 2	Contain the IPL records
3	Contains the volume label
4 or 5	Contain the CMS directory file
6	Contains the allocation map
7	Contains the alternate allocation map
8	Is the CMS level 1 pointer block
9 through 209	Are data blocks 1 through 201

The data blocks of the file created are now organized *sequentially* (blocks 9-209) after the blocks used by the CMS file system (blocks 1-8). Data block 1 is actually physical block 9, data block 2 is actually physical block 10, ..., and the last block (data block 201) is actually physical block 209.

Usage Notes

1. The RESERVE command is valid only for an accessed minidisk.
2. The RESERVE command does not rewrite the data blocks of the file being created. The data blocks contain whatever was left in the slot they occupy on disk. To clear these blocks with binary zeros, use the FORMAT command before the RESERVE command is issued.
3. If the disk specified is formatted with the RECOMP option, the RESERVE command ignores this option and assigns all cylinders or blocks to the file.
4. The block in the section above refers to a CMS physical block.

Responses

DMSRSV603R RESERVE will erase all files on disk *mode(vdev)*. Do you wish to continue? Enter 1 (YES) or 0 (NO).

To reply yes, enter 1 or 'YES'. To reply no, enter 0 or 'NO'. If you respond 'YES', you must *only* enter the character string 'YES'. You have indicated that a disk area is to be initialized; all existing files are erased. If the character string contains leading or trailing blanks, such as ' YES' or 'YES ', the response is processed as a 'NO' response. Responding 'NO', pressing the ENTER key, or entering a character string other than 'YES' cancels execution of the FORMAT command.

DMSRSV705I Disk remains unchanged

The response to continuing the execution of the RESERVE command was anything but YES.

DMSRSV733I Reserving disk *mode*

The disk represented by the mode letter *mode* is reserved. The response to continue the execution of the RESERVE command was YES or 1.

Messages and Return Codes

DMSRSV037E Filemode *mode (vdev)* is accessed as read/only [RC = 36]
 DMSRSV042E No fileid(s) specified [RC = 24]
 DMSRSV054E Incomplete fileid specified [RC = 24]
 DMSRSV069E Filemode *mode* not accessed [RC = 36]
 DMSRSV069E Output filemode *mode* not accessed [RC = 36]
 DMSRSV070E Invalid parameter *parameter* [RC = 24]
 DMSRSV109T Virtual storage capacity exceeded
 DMSRSV113S Disk(*vdev*) not attached [RC = 100]
 DMSRSV223E No filemode specified [RC = 24]
 DMSRSV260E Disk not properly formatted for RESERVE [RC = 16]
 DMSRSV908E File system error detected at virtual address *vdev*; reason code *nn* [RC = 100]
 DMSRSV909E Permanent I/O error on *vdev*; *csw = csw*, *sense = sense* [RC = 100]
 DMSRSV1264E Filemode *fm* is not associated with a minidisk [RC = 16]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

REVOKE AUTHORITY

Use the REVOKE AUTHORITY command to cancel the authorities that have been granted to other users for:

- one or more of your files in Shared File System (SFS) directories
- one of your SFS directories.

Format

REVoKe AUTHority	<pre> [<i>fn</i> <i>ft</i>] <i>dirid</i> FROM { <i>userid</i> * * { <i>nickname</i> PUBLIC ALL } } [(<i>options...</i>[])] Options: [KEEread] [TYPe NOType STACK [FIFO LIFO] LIFO FIFO] </pre>
-------------------------	---

Operands

fn ft

is the name and type of the file for which authority is to be removed. You must own the file to remove authorities. A special character (* or %) may be used to specify a set of files. See "Pattern Matching" on page 8 for a description of pattern matching using these characters.

dirid

is either:

- the name of the directory which contains the files from which authority is to be removed, if *fn ft* is specified.
- or
- the directory for which the authority is to be removed, if *fn ft* is not specified. You must be the owner of the directory to remove authorities. See "Naming Shared File System (SFS) Directories" on page 4 for a description of *dirid*.

FROM *userid*

FROM *nickname*

indicates the user or group of users from whom authority is being taken. Nicknames that have been set up through the NAMES command may be used for the *userid* or *nickname*.

FROM PUBLIC

revokes the PUBLIC authority given to all users who can connect to the file pool. This does not revoke any authorities that were granted individually.

FROM ALL

revokes authority from all users for a file or directory. If PUBLIC authority was granted, it is also revoked.

Options**KEEPread**

specifies that the authority should be changed from write to read. All aliases are kept in effect. If this option is not specified, all authority for the user is revoked from the file or directory and any aliases that the user has to the file are revoked.

Type

displays information at the terminal.

NOType

suppresses the display of information at the terminal. NOTYPE is the default.

STACK [FIFO]**STACK LIFO**

places the information in the console stack rather than displaying it at the terminal. FIFO is the default.

FIFO

specifies that the information is stacked in a first in, first out order. This option is equivalent to STACK FIFO.

LIFO

specifies that the information is stacked in a last in, first out order. This option is equivalent to STACK LIFO.

Usage Notes

1. To revoke authorities, you must be the owner of the file or directory; however, you may not revoke authority from yourself.
2. Revoking authority from an alias revokes the authority from the base file, if you own the base file.
3. You can revoke authority on a file or directory that you have locked UPDATE or EXCLUSIVE. A file or directory locked SHARE, or locked by another user, must be unlocked before revoking authority to it.
4. If a file is open, the revoke takes effect after the file is closed. To revoke authority from a directory, all files within the directory must first be closed. If you use special characters to specify a set of files, the directory cannot be open.
5. When you revoke authority from an accessed directory, the directory is released automatically from the revoked user and no message is issued.
6. If a revoke fails for a file specified by special characters, the processing continues with the next file name that matches the pattern.
7. If you wish to revoke authority to a user ID of ALL, PUBLIC, or any of the abbreviations of the PUBLIC operand (PUB, PUBL, and PUBLI), you need to create a nickname for that user ID using the NAMES command. Then use the nickname when issuing the REVOKE AUTHORITY command.
8. The REVOKE AUTHORITY command can only revoke authorities established through the GRANT AUTHORITY command. For any authorities granted through the External Security Manager (ESM), an ESM command must be used to revoke the authority.

Messages and Return Codes

- DMSJAU002E File *fn ft fm|dirname* not found [RC = 28]
- DMSJAU065E *option* option specified twice [RC = 24]
- DMSJAU066E *option1* and *option2* are conflicting options [RC = 24]
- DMSJAU1163E The REVOKE AUTHORITY command failed for *fn ft fm|dirname* [RC = 00]
- DMSJAU1184E File *fn ft fm|dirname* not found or you are not authorized for it [RC = 28]
- DMSJAU1184E Directory *dirname* not found or you are not authorized for it [RC = 28]
- DMSJAU1198E Directory *dirname* is currently open; it must be closed before you can change the authority to any file in it [RC = 70]
- DMSJAU1210E Directory *dirname* not found [RC = 28]
- DMSJAU1244W {User *userid*|At least one user in the list *userid*} was not granted READ|WRITE authority to *fn ft fm|dirname* [RC = 4]
- DMSJAU1245W Because *userid* owns *fn ft fm|dirname*, the authority cannot be revoked [RC = 4]
- DMSJAU1247W Public READ|WRITE authority did not previously exist on *fn ft fm|dirname* [RC = 4]
- DMSJAU1247W No users had {READ|WRITE} authority to *fn ft fm|dirname* [RC = 4]
- DMSJAU1248W Specified authorization revoked, but external security is still in effect for *fn ft fm|dirname* [RC = 4]
- DMSJAU1287W You do not own file *fn ft fm|dirname* [RC = 4]
- DMSJED069E Filemode *fm* not accessed [RC = 36]
- DMSJED109S Virtual storage capacity exceeded [RC = 104]
- DMSJED1187E Too many subdirectory levels in *dirid* [RC = 24]
- DMSJED1188E Filemode *fm* is not associated with a directory [RC = 74]
- DMSJED1189E Filemode *fm* is associated with a top directory [RC = 24]
- DMSJED1223E There is no default file pool currently defined [RC = 40]
- DMSJNL637E Missing nodeid for the AT operand [RC = 24]
- DMSJNL647E Userid not specified for *nickname* in *userid* NAMES file [RC = 32]
- DMSJNL647E Localid not specified for *userid* at *node* in *userid* NAMES file [RC = 32]
- DMSJNL653E Error executing *command* [RC = 40]
- DMSOUT1201E STACK option cannot follow FIFO or LIFO [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

RSERV

Use the RSERV command in CMS/DOS to copy, display, print, or punch a VSE relocatable module from a private or system library.

Format

RSERV	<i>modname</i> [<i>ft</i>] [(options... [])]
	TEXT
	<u>Options:</u> [DISK] [PRINT] [PUNCH] [TERM]

Operands

modname

specifies the name of the module on the VSE private or system relocatable library. The private library, if any, is searched before the system library.

ft

specifies the file type of the file to be created on your disk or directory accessed as A. "ft" defaults to TEXT if a file type is not specified. The file name is always the same as the module name.

Options

You may specify as many options as you wish on the RSERV command, depending on which functions you want to perform.

DISK

copies the relocatable module onto your disk or directory accessed as A. If no other options are specified, DISK is the default.

PRINT

prints the relocatable module on the virtual printer.

PUNCH

punches the relocatable module on the virtual punch.

TERM

displays the relocatable module at your terminal.

Usage Notes

1. If you want to copy modules from a private relocatable library, you must issue an ASSGN command for the logical unit SYSRLB and identify the library on a DLBL command line using the ddname IJSYSRL.

To copy modules from the system relocatable library, you must have entered the CMS/DOS environment specifying a mode letter on the SET DOS ON command line.
2. The RSERV command ignores the assignment of logical units, and directs output to the devices specified on the option list.

Responses

If you use the TERM option, the relocatable module is displayed at the terminal.

Messages and Return Codes

DMSRRV003E	Invalid option: <i>option</i> [RC=24]
DMSRRV004E	Module <i>module</i> not found [RC=28]
DMSRRV006E	No read/write A filemode accessed [RC=36]
DMSRRV070E	Invalid parameter <i>parameter</i> [RC=24]
DMSRRV089E	Open error code <i>nm</i> on SYSRLB [RC=36]
DMSRRV097E	No SYSRES volume active [RC=36]
DMSRRV098E	No module name specified [RC=24]
DMSRRV099E	CMS/DOS environment not active [RC=40]
DMSRRV105S	Error <i>nm</i> writing file <i>fn ft fm</i> on disk or directory [RC=100]
DMSRRV113S	Disk(<i>vdev</i>) not attached [RC=100]
DMSRRV411S	Input error code <i>nm</i> on SYSaaa [RC= <i>rc</i>]

RTNDROP

Use the RTNDROP command to undo the binding of callable services library (CSL) routines that were loaded by the RTNLOAD command.

Format

RTNDrop	$\{$ <i>namelist</i> $\}$ *	$[($ options... $)]$
	<u>options:</u>	
	$[$ User SYstem GRoup <i>grpname</i> $]$	$[$ TYpe <u>NOType</u> $]$

Operands

namelist

specifies the list of CSL routines that are to be dropped. These names are the “run names” given to the routines when they were loaded. Run names are either the original names, given when the routines were created, or alias names, specified when the routines were loaded using the RTNLOAD command.

If the USER, SYSTEM, or GROUP option is specified, all routines associated with each run name satisfying the specified option are dropped. If none of the three options is specified, only the most recently loaded routine associated with each run name is dropped.

*

specifies that all routines satisfying the specified option (USER, SYSTEM, or GROUP) are to be dropped. If none of these three options is specified, an error message is generated.

Options

User

specifies that only routines loaded with the USER attribute are to be dropped.

SYstem

specifies that only routines loaded with the SYSTEM attribute are to be dropped.

GRoup *grpname*

specifies that only routines loaded with the given *grpname* are to be dropped.

TYpe

indicates that an informational message is to be issued for each routine that is successfully dropped.

NOType

indicates that messages are not to be issued when routines are successfully dropped. This is the default.

Usage Notes

1. The maximum number of routines you can specify on *namelist* is limited only by programming language constraints and the type of terminal being used.
2. If you issue RTNDROP from the CSLLIST command area and specify any CSL routine names, a name list is built; the RTNDROP command will act upon any routine names you specify *plus* the routine shown on the CSLLIST screen.
3. The following commands are related to RTNDROP:
 - CSLLIST - displays a list of routines contained in a callable services library
 - RTNLOAD - loads a CSL routine
 - RTNMAP - lists the CSL routines that are currently loaded
 - RTNSTATE - determines the status of a specific CSL routine.

Examples

1. Issuing the following command


```
RTNDROP * (USER)
```

drops all routines that were loaded with the USER attribute.
2. Issuing the following command


```
RTNDROP GET (GROUP LIBXYZ)
```

drops any routines loaded with the run name GET and the group name LIBXYZ.
3. Issuing the following command


```
RTNDROP SINE COSINE TANGENT
```

drops only the most recently loaded routine associated with each of the three run names.

Messages and Return Codes

- DMSCRR065E *option* option specified twice [RC=24]
- DMSCRR066E *opt1* and *opt2* are conflicting options [RC=24]
- DMSCRR624W No CSL routines are loaded [RC=28]
- DMSCRR1086E Namelist is invalid: "*" is not valid with routine names [RC=24]
- DMSCRR1087E Either the USER, SYSTEM, or GROUP option must be specified if namelist is specified as "*" [RC=24]
- DMSCRR1088W Routine *rtname* cannot be dropped because it is not loaded [RC=4]
- DMSCRR1088W Routine *rtname* cannot be dropped because it was not loaded with the specified attribute [RC=4]
- DMSCRR1088W Routine *rtname* cannot be dropped because it was not loaded with the specified group name [RC=4]
- DMSCRR1089I *rtname* has been dropped
- DMSCRR1129W No routines were dropped [RC=4]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

RTNLOAD

Use the RTNLOAD command to search for, load, and bind a callable services library (CSL) routine to a fixed location in storage, making it available for invocation. Once a routine is loaded, it is known by either its original name or a user-specified alias. Using RTNLOAD allows an application program to use the same version of a routine no matter how many times the routine is called during a given execution of the program.

The routines either reside in a DASD-based callable service library or in a segment-based callable services library.

Format

RTNLoad	$\left\{ \begin{array}{l} \textit{namelist} \left[\left(\left[\begin{array}{l} \text{FRom } * \\ \text{FRom } \textit{library} \left[\begin{array}{l} \text{IN } \textit{fm} \\ \text{IN } \textit{dirid} \end{array} \right] \right] \left[\begin{array}{l} \text{ALias} \\ \text{NOAlias} \end{array} \right] \left[\text{other options...}(\text{ }) \right] \right] \right] \\ * \left(\text{FRom } \textit{library} \left[\begin{array}{l} \text{IN } \textit{fm} \\ \text{IN } \textit{dirid} \end{array} \right] \left[\text{NOAlias} \right] \left[\text{other options...}(\text{ }) \right] \right) \end{array} \right\}$ <p>Other options:</p> $\left[\begin{array}{l} \text{User} \\ \text{SYstem} \end{array} \right] \left[\text{GRoup } \textit{grpname} \right] \left[\begin{array}{l} \text{TYpe} \\ \text{NOType} \end{array} \right] \left[\begin{array}{l} \text{PUsh} \\ \text{NOPush} \end{array} \right]$
---------	--

Operands

namelist

specifies the list of CSL routines to be searched for, loaded into user storage if necessary, and bound. A list of routines consists of either individual routine names for the NOALIAS option, or routine name/alias name pairs for the ALIAS option. In the following example, the first line shows three routines being loaded without alias names, and the second line shows two routines being loaded with alias names.

```
rtnload rtn1 rtn2 rtn3 ... (NOAlias
```

```
rtnload rtn4 alias4 rtn5 alias5 ... (Alias
```

Once a routine is loaded, it is known by its "run name." The run name is either the original name given when the routine was created or, if specified, an alias name given when the routine was loaded.

*

specifies that all routines from the specified library are to be loaded. Alias names are not allowed with this operand.

Options

FROM *library*

specifies the library containing the routines to be loaded. The library *need not* be in the GLOBAL CSLLIB search order; FROM implies direct library specification.

Notes:

1. Saved segment resident libraries are searched first.
2. Libraries that can no longer be found are ignored.

FROM *

specifies that libraries should be searched using the library search order; the default callable services library VMLIB will be implicitly appended to the current search order. (See the GLOBAL command). This is the default.

IN *fm*

identifies the file mode of the accessed minidisk or SFS directory containing the library. This option is not valid if you have specified the library as an asterisk (*). The library *need not* be in the GLOBAL CSLLIB search order; IN implies direct library specification.

IN *dirid*

identifies the SFS directory containing the library. This option is not valid if you have specified the library as an asterisk (*). The library *need not* be in the GLOBAL CSLLIB search order; IN implies direct library specification. See "Naming Shared File System (SFS) Directories" on page 4 for a description of the different ways to specify *dirid*.

ALias

specifies that a routine name/alias name pair must be specified for each routine to be loaded. Once a routine is loaded, it is known by its "run name," which is either its original name or an alias name (if one was specified with the ALIAS option). A routine to be loaded under its original name can have an equal sign (=) specified for its alias name. This option is not valid if you specify *namelist* as an asterisk (*).

NOAlias

specifies that routines are loaded under their original names. This is the default.

User

specifies that the routines are to be dropped if an abend occurs. This is the default.

SYstem

specifies that these routines are to be retained if an abend occurs.

Note: When loading a CSL into a saved segment, you must specify the SYSTEM option on the SEGMENT LOAD command for CSL routines to survive abend processing—even if you specified the SYSTEM option on RTNLOAD when loading individual CSL routines.

GRoup *grpname*

identifies the name of a collection of related routines.

TYpe

indicates that an informational message is to be issued for each routine that is successfully loaded.

NOType

indicates that messages are not to be issued when routines are loaded. This is the default.

PUSH

specifies that if a routine of the same run name is already loaded, its address is saved during this load. Subsequent invocation of this routine executes the most recently loaded version. A RTNDROP of this run name drops the most recently loaded version and makes the most recently saved version the current loaded version. If the routine is not already loaded, this option is ignored. This is the default.

NOPush

specifies that if a routine with a specified run name is already loaded, no action is taken. A warning message is issued to state that the routine is already loaded.

Usage Notes

1. Routine names, library names, and alias names can each be from one to eight characters. The valid characters are A-Z, a-z, 0-9, \$, #, @, +, - (hyphen), : (colon), and _ (underscore).
2. A namelist can specify a maximum of 256 routine names.
3. Alias names are useful because they let you have duplicate routine names.
If the ALIAS option is specified, *namelist* must not be an asterisk (*), and it must have an even number of entries. An equal sign (=) can also be used for an alias name.
4. Several routines can be loaded under the same *grpname*, and can be dropped by issuing a single RTNDROP command for the *grpname*.
5. If the IN *dirid* option is specified, the FROM *library* option must be specified and the library cannot be specified as an asterisk (*).
6. If the "FROM **" option is specified, the *namelist* must not be specified as an asterisk (*).
7. The following commands are related to RTNLOAD:
 - If you issue RTNLOAD from the CSLLIST command area and specify any CSL routine names, a name list is built; the RTNLOAD command will act upon any routine names you specify *plus* the routine shown on the CSLLIST screen.
 - CSLLIST - displays a list of routines contained in a callable services library
 - RTNMAP - lists the CSL routines that are currently loaded
 - RTNDROP - drops a loaded CSL routine
 - RTNSTATE - determines the status of a specific CSL routine.
8. If you specify a directory name in the IN *dirid* option and the directory is not accessed, the directory is temporarily accessed as the next available file mode letter. When the RTNLOAD command is complete, the file mode is released.

Examples

1. `rtnload * (from mylib`
loads all of the routines from the callable services library called MYLIB. RTNLOAD first searches saved segments to find MYLIB, and if unsuccessful, then follows the CMS search order.
2. `rtnload sample xyzy (from mylib alias`

indicates that the routine *SAMPLE* is to be loaded from the callable services library *MYLIB*, and that it is to be known as *XYZZY* after it is loaded.

3. *rtnload sine (push)*
 finds the first *SINE* routine in a library in the *CSL* search order and binds the *SINE* routine to the address where it is now loaded. The address of the current *SINE* routine is preserved.
4. *rtnload cosine (from yourlib in node7:johndoe.jd.test)*
 finds the *COSINE* routine in *YOURLIB* on the specified directory and binds it to the address where it is now loaded.
5. *rtnload fileread (from mylib group project1)*
 loads the routine *FILEREAD* from the callable services library *MYLIB*, and establishes its group name to be *PROJECT1*.

Messages and Return Codes

- DMSCAX1089I rtnname* has been loaded [RC=0]
- DMSCAX1099W rtnname* has already been loaded [RC=4]
- DMSCRL065E option* option specified twice [RC=24]
- DMSCRL066E opt1* and *opt2* are conflicting options [RC=24]
- DMSCRL639E* Error in *rtnname* routine; return code was *retcode*
- DMSCRL1084E ALIAS* option is not valid if namelist is specified as "*" [RC=24]
- DMSCRL1084E ALIAS* option is not valid if namelist has an odd number of entries [RC=24]
- DMSCRL1085E* A library name must be specified if namelist is specified as "*" [RC=24]
- DMSCRL1085E* A library name must be specified if the *IN* option is specified [RC=24]
- DMSCRL1086E* Namelist is invalid: "*" is not valid with routine names [RC=24]
- DMSCRL1086E* Namelist is invalid: more than 256 names are specified [RC=24]
- DMSCRL1090E* Invalid routine name *rtnname* specified [RC=24]
- DMSCRL1097E* Routine *rtnname* not found [RC=8]
- DMSCRL1098E* None of the specified routines were found [RC=8]
- DMSCRL1100E* No filemode is available to access *dirid* [RC=12]
- DMSCRL1136E* Unable to gain access to library *libname* [RC=28]
- DMSCRL1136W* Unable to gain access to library *libname* [RC=4]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

RTNMAP

Use the RTNMAP command to display information about callable services library (CSL) routines that are currently loaded and bound to an address.

Format

RTNMap	<pre> { <i>runname</i> * } [(options... [])] Options: [STACK [FIFO] STACK LIFO FIFO LIFO] [User SYstem] [GRoup <i>grpname</i>] [Header NOHeader] [ALL] </pre>
---------------	---

Operands

runname

is the run name of the CSL routine to be listed.

Note: This run name is not necessarily the routine's original name; if an alias was specified when the routine was loaded, the alias name must be used.

*

indicates that information is to be shown about the set of routines that satisfies the specified options. (These command options are described below.)

Options

STACK [FIFO]

STACK LIFO

places the output on the program stack, one line at a time, instead of displaying it at the terminal. The information is stacked either FIFO (first-in-first-out) or LIFO (last-in-first-out). The default order is FIFO.

FIFO

places the output on the program stack, one line at a time, in FIFO order. This option is equivalent to STACK and STACK FIFO.

LIFO

places the output on the program stack, one line at a time, in LIFO order. This option is equivalent to STACK LIFO.

User

specifies that information is displayed only for routines loaded with the USER option (on the RTNLOAD command). See also usage note 1.

System

specifies that information is displayed only for routines loaded with the SYSTEM option (on the RTNLOAD command). See also usage note 1.

Group *grpname*

specifies that only routines that were loaded into the specified *grpname* (on the RTNLOAD command) are to be displayed. If either USER or SYSTEM is also specified, information will be displayed about a routine only if it meets both sets of criteria.

Hheader

specifies that the output at the user's terminal is to be preceded by a header that identifies the columns of output displayed. See an example of this header in "Responses" below. HEADER is the default, unless one of the stack options (STACK, FIFO, or LIFO) is specified; in that case the HEADER option is not allowed.

NOHeader

specifies that the output at the user's terminal is not to be preceded by a header to identify the columns of displayed output.

When one of the stack options (STACK, FIFO, or LIFO) is specified, the NOHEADER option is the default and the HEADER option is not allowed.

All

specifies that information is displayed about all routines that meet the specified criteria, even if the routines were reloaded with conflicting options using RTNLOAD ... (PUSH. If ALL is not specified, information is displayed only for the most recently loaded copy of the routine(s) specified by *name* or *.

Inactive routines are preceded by an "*" in the output; active routines are preceded by a blank in the output.

Usage Notes

1. If neither the USER nor SYSTEM option is specified, RTNMAP displays information according to the routine(s) specified in the *name* or * operand and, if specified, the GROUP and ALL options.
2. If a routine was loaded with multiple alias names and RTNMAP * is specified, a line is displayed for each alias name.
3. The following commands are related to RTNMAP:
 - CSLLIST - displays a list of routines contained in a callable services library
 - RTNLOAD - loads a CSL routine
 - RTNDROP - drops a loaded CSL routine
 - RTNSTATE - determines the status of a specific CSL routine.

Examples

1. A routine named FORMULA exists in a callable services library named TEST. It was loaded with an alias name FORMULA2 using the following command:

```
RTNLOAD FORMULA FORMULA2 (FROM TEST ALIAS
```

Suppose the routine took up 300 bytes of storage and it has been invoked three times since it was loaded. To display information about this routine, entering the command

```
RTNMAP FORMULA2 (USER HEADER
```

produces the following output:

Alias	Name	Library	UseCount	LoadAddr	Size	Attrib.	Group
FORMULA2	FORMULA	TEST	3	005CBC68	300	USER	

2. Suppose the FORMULA routine was reloaded with the same alias name, but with the SYSTEM option, using the following command:

```
RTNLOAD FORMULA FORMULA2 (FROM TEST ALIAS SYSTEM
```

This copy of the routine also takes up 300 bytes of storage and it has been invoked two times since it was loaded. Entering the command

```
RTNMAP FORMULA2 (ALL HEADER
```

produces the following output:

Alias	Name	Library	UseCount	LoadAddr	Size	Attrib.	Group
FORMULA2	FORMULA	TEST	2	005CBC68	300	SYSTEM	
*FORMULA2	FORMULA	TEST	3	005CBC68	300	USER	

Responses

RTNMAP produces output in the following format (when the HEADER option is on):

Alias	Name	Library	UseCount	LoadAddr	Size	Attrib.	Group
rtnalias	rtnname	libname	count	address	size	attrib	grpname

where:

rtnalias

is the name used to invoke this routine. If the name is preceded by an asterisk (“*”), a copy of this routine was reloaded using the PUSH option on RTNLOAD; the information displayed after the “*” pertains to a previously-loaded copy of the routine, not the copy that can be currently invoked.

rtnname

is the original name of this routine as it resides in the library.

libname

is the name of the library from which this routine was loaded.

count

is the number of times that this routine has been invoked via DMSCSL since it was loaded. This value can be used to determine whether to RTNDROP a routine or whether to move the library into a segment. If the routine is called via the CSLCALL macro, count is not incremented. Also, CMS commands that use CSLCALL rather than DMSCSL, will not update the counter.

address

is the address where the routine was loaded. (If you are debugging a new CSL routine, this address will be useful.)

size

shows how much user storage (in bytes) is occupied by this routine. A routine from a CSL that resides in a saved segment is shown with a size of 0 (zero); this is because such a routine does not occupy user storage and no additional storage is made available if RTNDROP is issued for this routine.

attrib

shows whether the routine was loaded with the USER attribute (the default) or with the SYSTEM attribute. If an abend occurs, routines with the SYSTEM attribute will be retained, while routines with the USER attribute will be dropped. (For example, an HX command results in an abend.) The attribute is set by the RTNLOAD command.

grpname

is the name of the group that the routine was loaded into using the RTNLOAD command.

Messages and Return Codes

DMSCRM065E *option* option specified twice [RC = 24]

DMSCRM066E *opt1* and *opt2* are conflicting options [RC = 24]

DMSCRM1088E Routine *rtname* cannot be mapped because it is not loaded [RC = 28]

DMSCRM1088E Routine *rtname* cannot be mapped because it was not loaded with the specified attribute [RC = 28]

DMSCRM1088E Routine *rtname* cannot be mapped because it was not loaded with the specified group name [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

RTNSTATE

Use RTNSTATE to verify the existence of a loaded callable services library (CSL) routine.

Format

RTNState	$\left\{ \begin{array}{c} \textit{runname} \\ * \end{array} \right\} \quad [(\textit{options... } [])]$ <p>Options:</p> $\left[\begin{array}{c} \text{User} \\ \text{SYstem} \end{array} \right] \quad [\text{GRoup } \textit{grpname}]$
-----------------	--

Operands

runname

is the name of a specific CSL routine whose load status is being verified.

Note: This run name is not necessarily the routine's original name; if an alias was specified when the routine was loaded, the alias name must be used.

*

supplies load status information about all CSL routines, or the subset of routines that satisfies the specified options.

Options

User

When used with a specific routine name, this indicates that a non-zero return code should be given, unless the named routine was loaded with the USER attribute. When used with an "*" for the name, this indicates that a non-zero return code should be given, unless at least one routine was loaded with the USER attribute.

SYstem

When used with a specific routine name, this indicates that a non-zero return code should be given, unless the named routine was loaded with the SYSTEM attribute. When used with an "*" for the name, this indicates that a non-zero return code should be given, unless at least one routine was loaded with the SYSTEM attribute.

GRoup *grpname*

When used with a specific routine name, this indicates that a non-zero return code should be given, unless the named routine was loaded into the given group. When used with an "*" for the name, this indicates that a non-zero return code should be given, unless at least one routine was loaded into the given group.

Usage Notes

1. This command is intended to be used from an exec.
2. The output of RTNSTATE is a return code. See the following examples for more information.
3. The following commands are related to RTNSTATE:
 - CSLLIST - displays a list of routines contained in a callable services library
 - RTNLOAD - loads a CSL routine
 - RTNDROP - drops a loaded CSL routine
 - RTNMAP - lists the CSL routines that are currently loaded.

Examples

1. RTNSTATE MYRTN gives the following return codes:
 - 0 if the routine was loaded
 - 28 if the routine was not loaded.
2. RTNSTATE MYRTN (USER) gives the following return codes:
 - 0 if the routine was loaded with the USER attribute
 - 4 if the routine was loaded with the USER attribute and then re-loaded with the SYSTEM attribute and the PUSH option
 - 8 if the routine was loaded with the SYSTEM attribute but was not loaded with the USER attribute
 - 28 if the routine was not loaded.
3. RTNSTATE MYRTN (SYSTEM) gives the following return codes:
 - 0 if the routine was loaded with the SYSTEM attribute
 - 4 if the routine was loaded with the SYSTEM attribute and then re-loaded with the USER attribute and the PUSH option
 - 8 if the routine was loaded with the USER attribute but was not loaded with the SYSTEM attribute
 - 28 if the routine was not loaded.
4. RTNSTATE MYRTN (GROUP MYGROUP) gives the following return codes:
 - 0 if the routine was loaded into MYGROUP
 - 4 if the routine was loaded into MYGROUP and then re-loaded into another group with the PUSH option
 - 8 if the routine was loaded but not into MYGROUP
 - 28 if the routine was not loaded.
5. RTNSTATE MYRTN (USER GROUP MYGROUP) gives the following return codes:
 - 0 if the routine was loaded into MYGROUP with the USER attribute
 - 4 if the routine was loaded into MYGROUP with the USER attribute and then re-loaded into another group with the PUSH option, or if the routine was loaded into MYGROUP with the USER attribute and then re-loaded with the SYSTEM attribute and the PUSH option
 - 8 if the routine was loaded but not into MYGROUP with the USER attribute
 - 28 if the routine was not loaded.
6. RTNSTATE * gives the following return codes
 - 0 if any routines are loaded
 - 28 if no routines are loaded.
7. RTNSTATE * (USER) gives the following return codes:

- 0 if at least one routine was loaded with the USER attribute AND no routines that were loaded with the USER attribute have been re-loaded with the SYSTEM attribute and the PUSH option
 - 4 if at least one routine was loaded with the USER attribute and then re-loaded with the SYSTEM attribute
 - 28 if no routines were loaded with the USER attribute.
8. RTNSTATE * (USER GROUP MYGROUP) gives the following return codes:
- 0 if at least one routine was loaded into MYGROUP with the USER attribute AND no routines were loaded into MYGROUP with the USER attribute and then reloaded into another group with the PUSH option or reloaded with the SYSTEM attribute and the PUSH option
 - 4 if one or more routines were loaded into MYGROUP with the USER attribute and then reloaded into a different group with the PUSH option or reloaded with the SYSTEM attribute and the PUSH option
 - 28 if no USER routines were loaded into MYGROUP.

Messages and Return Codes

DMSCRS003E Invalid option: *option* [RC=24]
 DMSCRS065E *option* option specified twice [RC=24]
 DMSCRS066E *opt1* and *opt2* are conflicting options [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

RUN

Use the RUN command to initiate a series of functions on a file depending on the file type. The RUN command can select or combine the procedures required to compile, load, or start execution of the specified file.

Format

RUN	<i>fn</i> [<i>ft</i> [<i>fm</i>]] [(<i>args...</i>)]
------------	---

Operands

fn

is the file name of the file to be manipulated.

ft

is the file type of the file to be manipulated. If file type is not specified, a search is made for a file with the specified file name and the file type of EXEC, MODULE, or TEXT (the search is performed in that order). If the file type of an input file for a language processor is specified, the language processor is invoked to compile the source statements and produce a TEXT file. If no compilation errors are found, LOAD and START may then be called to initiate program execution. The valid file types and resulting action for this command are:

File type	Action
EXEC	The EXEC processor is called to process the file.
MODULE	The LOADMOD command is issued to load the program into storage and the START * command begins execution of the program at the default entry point.
TEXT	The LOAD command brings the file into storage in an executable format and the START * command executes the program beginning at the default entry point.
FORTRAN	The FORTRAN processor module that is called is FORTVS, FORTRAN, FORTGI, GOFORT, or FORTHX, whichever is found first. Object text successfully compiled by the FORTGI or FORTHX processors will be loaded and executed.
TESTFORT	The TESTFORT module is called to initiate FORTRAN Interactive Debug and will process a TEXT file that has been compiled with the TEST option.
FREEFORT	The GOFORT module is called to process the file.
COBOL	The COBOL processor module that is called is COBOL or TESTCOB, whichever is found first. After successful compilation, the program text will be loaded and executed.
PLI PLIOPT	The PLIOPT processor module is called to process the file. After successful compilation, the program text will be loaded and executed.

fm

is the file mode of the file to be loaded by the LOADMOD command. If *fm* is specified, a file type must also be specified. The *fm* field is only beneficial when attempting to execute a module. All other functions use the default search order to locate a file on one of your disks or directories.

args

are arguments you want to pass to your program. You can specify up to 13 arguments in the RUN command, provided they fit on a single input line. Each argument is left-justified, and any argument more than eight characters long is truncated from the right.

Usage Notes

1. If you are executing a CMS EXEC or EXEC 2 file, the arguments you enter on the RUN command line are assigned to the variable symbols &1, &2, and so on. If you are executing a System Product Interpreter program, the arguments that you enter on the RUN command line are available via the arg instruction.

The RUN command passes only the file name (*fn*) of an exec to the EXEC processor. Therefore, you cannot use "fm" to select a particular exec.

2. Before using the RUN command, you should issue the GLOBAL command to identify the required libraries.
3. If you are executing a TEXT or MODULE file, or compiling and executing a program, the arguments are placed in a parameter list and passed to your program when it executes. The arguments are placed in a series of doublewords in storage, terminated by X'FF'. If you enter:

```
run myprog (charlie dog
```

the arguments *, CHARLIE, and DOG are placed in doublewords in a parameter list, and the address of the list is in register 1 when your program receives control.

Note: You cannot use the argument list to override default options for the compilers or for the LOAD or START commands.

4. The RUN command is not designed for use with CMS/DOS.
5. The RUN command cannot be used for COBOL and PL/I programs that require facilities not supported under CMS. For specific language support limitations, see *VM/SP Planning Guide and Reference*.
6. If you want to issue RUN from an exec program, you should precede it with the EXEC command; that is, specify

```
exec run
```

Responses

Any responses are from the programs or procedures that executed within the RUN command.

Messages and Return Codes

DMSRUN001E	No filename specified [RC = 24]
DMSRUN002E	File <i>fn ft fm</i> not found [RC = 28]
DMSRUN048E	Invalid mode <i>fm</i> [RC = 24]
DMSRUN070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSRUN999E	No <i>ft</i> module found [RC = 28]

RUN

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in the Shared File System	813

SEGMENT

Use the SEGMENT command to manage saved segments. The SEGMENT command allows you to:

- reserve CMS storage for a saved segment that will reside in your virtual machine
- assign a logical saved segment to a physical saved segment
- load a saved segment
- detach a saved segment
- release storage previously reserved for a saved segment.

The SEGMENT command uses DIAGNOSE X'64' to manage these segments. The following pages describe in detail the five functions of the SEGMENT command:

- SEGMENT ASSIGN
- SEGMENT LOAD
- SEGMENT PURGE
- SEGMENT RELEASE
- SEGMENT RESERVE.

There is also a SEGMENT macro. For reference information on the SEGMENT macro, see the *VM/SP Application Development Reference for CMS*. For usage information on both the SEGMENT command and macro, see the *VM/SP Application Development Guide for CMS*.

SEGMENT ASSIGN

Use the SEGMENT ASSIGN command to change the physical segment from which a logical segment will be used.

Format

SEGMENT ASSIGN	<i>lsegname</i> <i>psegname</i>
-----------------------	---------------------------------

Operands

lsegname

specifies the 1-8 character name of the logical segment.

psegname

specifies the 1-8 character name of the physical segment.

Usage Note

None

Example

You have two logical segments with the same name, EXECS, residing in two physical segments, PSEG1 and PSEG2. If you want to use the EXECS segment in PSEG2, you would issue the command,

```
segment assign execs pseg2
```

Then, whenever you issue another one of the SEGMENT commands or macro, such as SEGMENT LOAD, the EXECS segment in PSEG2 is used.

Messages and Return Codes

DMSDCT109S Virtual storage capacity exceeded, return code *rc* from storage management [RC = 104]

DMSDCT1274E Logical segment *lsegname* does not exist in physical segment *psegname*. [RC = 28]

DMSDCT1275E Logical segment *segname* is currently active and cannot be assigned. [RC = 36]

DMSDCT1277E Logical segment *segname* does not exist. [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SEGMENT LOAD

Use the SEGMENT LOAD command to load a saved segment.

Format

SEGMENT LOAD	<i>segname</i> [(options...[])] <u>Options:</u> [SYSTEM] [SHare] [USER] [NOSHare]
---------------------	--

Operands

segname

specifies the 1-8 character name of:

- a logical saved segment
 - a physical saved segment
- or
- the segment space reserved if you used the SEGMENT RESERVE command.

Note: See "SEGMENT RESERVE" for a description of segment spaces.

Options

SYSTEM

specifies that the segment space that contains the saved segment is not to be released during ABEND processing. If you do not specify SYSTEM and ABEND recovery is invoked, the segment space will be released.

If you already reserved segment space with the SEGMENT RESERVE command for a physical segment, the SYSTEM option specified on the SEGMENT RESERVE command overrides the values specified on the SEGMENT LOAD command.

Note: If your saved segment is to include a callable services library (CSL), and if any individual CSL routines were loaded with the SYSTEM option on the RTNLOAD command, you must still specify the SYSTEM option on SEGMENT LOAD for the CSL routines to survive abend processing.

USER

specifies that the segment space is to be released during ABEND processing. This is the default option.

SHare

loads a shared copy of the saved segment. This is the default value.

NOSHare

loads a non-shared copy of the saved segment.

Note: To use the NOSHARE option on VM/XA, users who issue SEGMENT

LOAD must also have an appropriate NAMESAVE entry in their user directory entry.

Usage Notes

1. You can load saved segments even if they reside within a virtual machine's address space. However, the CMS storage must be free storage; that is, no programs or data may already reside there.
2. The **SEGMENT LOAD** command is an alternative to the **DIAGNOSE X'64'** instruction. The **SEGMENT** command uses the following process to locate the saved segment it loads:
 - a. CMS searches the list of logical saved segments for one with the same name as that specified on the **SEGMENT LOAD** macro. If one is found, space for the associated physical saved segment is reserved (if necessary), the physical saved segment or segment space is loaded (if necessary), and the logical saved segment contents are processed.
 - b. CMS searches the list of storage areas reserved with the **SEGMENT** command to determine if an area has been reserved for a saved segment with the requested name. If one is found, the physical saved segment or segment space is loaded (if necessary). If the specified name is a logical saved segment, the contents are processed.
 - c. If no reserved storage area exists, CMS issues a **CP FINDSYS** to determine whether the requested segment is a physical saved segment or a segment space. If so, a **SEGMENT RESERVE** command is issued to create a reserved area and the physical saved segment or segment space is loaded.
 - d. If the requested segment is not a reserved area, physical saved segment, or segment space, the appropriate return code (**RC = 44**) is returned to the calling program.

This process allows an application to be loaded even if the segment resides within the virtual machine.

3. When a logical segment is loaded,
 - programs in it are established as nucleus extensions or subcommand processors
 - Execs are established as execs-in-storage
 - CSL libraries are made usable by the **GLOBAL CSLLIB** command
 - language messages repositories are processed
 - user exit routines are called.Objects in other logical segments in the physical segment are not processed.
4. The language of application message repositories must match the current system language to be added to the active set of applications.
5. Nucleus extensions, subcommand processors, and execs established by **SEGMENT LOAD** override objects of the same name. Nucleus extensions in segments can be dropped (see **NUCXDROP** command), bringing back the previous definition of the name. Purging nucleus extensions and subcommand processors will also bring the previous definition back into effect.
6. The **SHARE** attribute of a physical segment that contains logical segments is set by the first logical segment loaded in the physical segment. If other logical segments are loaded in the physical segment, they cannot change the **SHARE** attribute. All logical segments in the same physical segment have the same **SHARE** attribute. If you specify a **SHARE** or **NOSHARE** option that does not

match the SHARE attribute of the physical segment, the segment is not loaded and a non-zero return code is returned.

Messages and Return Codes

- DMSDCT109S Virtual storage capacity exceeded, return code *rc* from storage management [RC = 104]
- DMSDCT343E Storage in range *address-address* for *name* in use. [RC = 41]
- DMSDCT1083E Saved segment *segname* does not exist [RC = 44]
- DMSDCT1266E Error occurred while loading logical segment *segname*, return code *rc* [RC = 256]
- DMSDCT1267E Error occurred while loading user objects, return code *rc* [RC = 256]
- DMSDCT1270E The SHARE/NOSHARE option specified does not match the SHARE attribute of the containing physical segment. [RC = 36]
- DMSDCT1271E *name* contains reserved and/or loaded logical segments and cannot be reserved, loaded, or purged. [RC = 36]
- DMSDCT1272E Physical segment *segname* is already active. [RC = 36]
- DMSDCT1295E *name* segment space contains reserved or loaded member saved segments and cannot be reserved or loaded. [RC = 36]
- DMSDCT1296E *name* member saved segment cannot be reserved or loaded in a segment space that is already reserved or loaded. [RC = 36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

RC	Meaning
0	Normal
12	The saved segment exists and has already been loaded. The segment has not been reloaded.
44	Segment does not exist.
256	Error processing contents of logical segment.

You may receive other return codes from DIAGNOSE X'64'.

SEGMENT

SEGMENT PURGE

Use the SEGMENT PURGE command to purge a saved segment from a virtual machine.

Format

SEGMENT PURGE	<i>segname</i>
----------------------	----------------

Operands

segname

specifies the 1-8 character name of the segment to be purged.

Usage Notes

1. If SEGMENT LOAD reserved space for a saved segment, then SEGMENT PURGE automatically releases that space.
2. If the segment getting purged is a logical segment, all the objects in the segment are purged:
 - user exit routines are called
 - nucleus extensions and execs are dropped
 - subcommand processors are cleared
 - message repositories are deleted
 - libraries are removed from the list of callable services libraries.

If the logical segment is the only active segment in the physical segment, the physical segment is removed from the virtual machine.

Messages and Return Codes

- DMSDCS1083E Saved segment *segname* does not exist [RC=44]
DMSDCT109S Virtual storage capacity exceeded, return code *rc* from storage management [RC=104]
DMSDCT345E *segname* was not loaded via SEGMENT LOAD function [RC=40]
DMSDCT1268E Error occurred while purging logical segment *segname*, return code *rc* [RC=256]
DMSDCT1269E Error occurred while purging user object *name*, return code *rc* [RC=256]
DMSDCT1271E *name* contains reserved and/or loaded logical segments and cannot be reserved, loaded, or purged. [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

RC	Meaning
0	Normal

| **40** Segment is not loaded.

| **44** Segment does not exist.

| **256** Error processing contents of logical segment.

| You may receive other return codes from DIAGNOSE X'64'.

SEGMENT RELEASE

Use the SEGMENT RELEASE command to return a segment space reserved for a segment to CMS.

Format

SEGMENT RELEASE	<i>name</i>
------------------------	-------------

Operands

specifies the 1-8 character name of a segment space, physical segment, or logical segment for which corresponding storage is released.

Usage Note

1. If the specified name is a logical segment, the logical segment is removed from the list of reserved logical segments. If the physical segment that contains the logical segment no longer has any logical segment reserved or loaded, then the physical segment is removed from your virtual machine and the reserved storage is returned.

If the specified name is a physical segment, then all the loaded or reserved logical segments within the physical segment's space are released. Then the physical segment is released.

If the specified name is a segment space, all of the physical segments of this space are released as described in the previous step. Then the segment space is released.

You can reserve or load several logical segments within a physical segment, and then reclaim all the storage used by these segments by issuing the SEGMENT RELEASE command for the physical segment.

Messages and Return Codes

DMSDCS109S	Virtual storage capacity exceeded, return code <i>rc</i> from storage management [RC=104]
DMSDCS344E	Segment space <i>name</i> has not been reserved [RC=40]
DMSDCS345E	<i>segname</i> was not loaded via SEGMENT LOAD function [RC=40]
DMSDCS1083E	Saved segment <i>segname</i> does not exist [RC=44]
DMSDCS1268E	Error occurred while purging logical segment <i>segname</i> , return code <i>rc</i> [RC=256]
DMSDCS1269E	Error occurred while purging user object <i>name</i> , return code <i>rc</i> [RC=256]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SEGMENT RESERVE

Use the SEGMENT RESERVE command to create a segment space so that a subsequent SEGMENT LOAD command or macro, or a DIAGNOSE X'64', may be safely issued. SEGMENT RESERVE does not actually load a saved segment into storage.

Format

SEGMENT RESERVE	<i>name</i> [(options...[])] <u>Options:</u> [SYSTEM] [USER]
------------------------	---

Operands

name

specifies the 1-8 character name of a segment space, physical segment, or logical segment.

Options

SYSTEM

specifies that this segment space is not to be released during ABEND processing. If this option is not specified, the segment space will be released if ABEND recovery is invoked.

USER

specifies that this segment space is to be released during ABEND processing. This is the default option.

Usage Notes

1. The location of the segment space is determined by the location of the corresponding physical segment defined in CP. If *name* is a logical segment, the physical segment is the physical segment to which *name* is assigned.

Messages and Return Codes

DMSDCS109S	Virtual storage capacity exceeded, return code <i>rc</i> from storage management [RC = 104]
DMSDCS358E	Saved segment <i>name</i> has already been reserved [RC = 4]
DMSDCS1083E	Saved segment <i>segname</i> does not exist. [RC = 44]
DMSDCS1270E	The SHARE NOSHARE option specified does not match the SHARE attribute of the containing physical segment [RC = 24]
DMSDCS1083E	Saved segment <i>segname</i> does not exist [RC = 44]
DMSDCS1271E	<i>name</i> contains reserved and/or loaded logical segments and cannot be reserved, loaded, or purged. [RC = 36]
DMSDCS1272E	Physical segment <i>segname</i> is already active. [RC = 36]
DMSDCS1295E	<i>name</i> segment space contains reserved or loaded member saved segments and cannot be reserved or loaded. [RC = 36]
DMSDCS1296E	<i>name</i> member saved segment cannot be reserved or loaded in a segment space that is already reserved or loaded. [RC = 36]

SEGMENT

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

RC	Meaning
0	Normal
24	Segment space <i>name</i> already reserved for another segment.
40	Segment is not loaded.
44	Segment does not exist.
256	Error processing contents of logical segment.

Other returns codes may be issued by DIAGNOSE code X'64'.

SENDFILE

Use the SENDFILE command to send files or notes to one or more computer users on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network.

SENDFILE is one of several commands that references a “userid NAMES” file. By setting up a names file, you can identify recipients just by using nicknames, which are automatically converted into node and user ID. For information on creating a names file, see the NAMES command.

Format

SENDFile SFile	<pre>[fn ft [fm] [[TO] name...] [(options... [])]]</pre> <p>Options:</p> <pre>[Ack] [Filelist] [Log] [NEw] [NOTE]</pre> <pre>[NOAck] [NOFilelist] [NOLog] [Old]</pre> <pre>[Type]</pre> <pre>[NOType]</pre>
---------------------------------	---

Operands

fn

is the file name of the file to be sent.

ft

is the file type of the file to be sent.

fm

is the file mode of the file to be sent. If “*” is specified (the default), all accessed disks and directories are searched, and the first file found is sent. This operand can be omitted if the first “name” would not be misinterpreted as a file mode, or if the keyword “TO” is used.

TO

is a keyword operand. It can be omitted if the first “name” is not “TO”.

name

is one or more “names” of the computer users to whom the file is to be sent. If the same recipient is specified more than once, he receives only one copy of the file. The “name” may take any of the following forms, and the different forms can be freely intermixed:

- a “nickname” that can be found in the file “userid NAMES,” where “userid” is your user ID. This nickname may represent a single person (on your computer or on another computer), or a list of several people. See the NAMES command for more information on nicknames.
- a user ID of a computer user on your computer. If a name cannot be found in the “userid NAMES” file, it is assumed to be a user ID of someone on your computer.

- “userid AT node,” which identifies a user (“userid”) on your computer or another computer (“node”).

You cannot send files to a user ID named “AT” or “CC:”.

If no operands are specified, a menu is displayed. This menu is described in the Usage Note below, “Using the SENDFILE Menu.”

Options

Ack

requests an acknowledgment be returned to you and logged when the recipient receives your file (using the RECEIVE command). Acknowledgments are added to your “userid NETLOG” file. An acknowledgment is sent only if the NEW option is also in effect.

NOAck

requests that no acknowledgment be returned when the recipient RECEIVES a file.

Filelist

specifies that the file (fn ft fm) is a list of files in the format of a CMS exec file produced by the LISTFILE command issued with the EXEC option, or a file saved from a FILELIST command. This option is used to send multiple files with only one invocation of SENDFILE. Both the file containing the list of files and each file in the list are sent.

Lines beginning with an asterisk (*) and blank lines are ignored. All exec tokens (for example, &1, &2) or any token beginning with an ampersand (&) is ignored.

For information on creating a list of files that can be saved and used to send multiple files, see the FILELIST command, the usage note “Saving a List of Files.”

NOFilelist

specifies that the file is not a list of files.

Log

specifies that the recipients, date, and time of this file transmission are logged in a file called “userid NETLOG.” This log is updated when acknowledgments of sent files are received (if they were requested). Do not use this option if you have no read/write disk or directory accessed.

NOLog

specifies that this file transmission is not to be logged.

NEw

specifies that header records are added and the file is sent as described below, in “Format of the File Sent by SENDFILE.” If this option is specified, the recipient *must use RECEIVE* to read the file.

Old

specifies that the file is sent using DISK DUMP. This option should be specified when the recipient of the file does not have the RECEIVE command available to read the file. When OLD is specified, no acknowledgment (the ACK option) can be requested.

NOTE

specifies that the file is to be sent as a note (that was prepared using the NOTE command). The “TO” operand and the list of names cannot be specified if this option is given. If no file is specified, the file “userid NOTE *” is sent as a note.

(On a display terminal, the PF5 key is set to this option in the NOTE command environment.)

Type

specifies that the files sent and the user IDs and nodes to which the files were sent are displayed at the terminal.

NOType

specifies that no information is to be displayed.

Usage Notes

1. Tailoring the SENDFILE Command Options

You can use the DEFAULTS command to set up options and/or override command defaults for SENDFILE. However, the options you specify in the command line when entering the SENDFILE command override those specified in the DEFAULTS command. This allows you to customize the defaults of the SENDFILE command, yet override them when you desire. Refer to the DEFAULTS command description for more information.

2. Using the SENDFILE Menu (Display Terminals Only)

Enter the SENDFILE command without operands to display a menu, on which you “fill in the blanks” with the necessary information. A sample SENDFILE menu is shown in the “Examples,” below.

The File Identifier:

You type the file name, file type, and file mode of a file that you want to send directly on the menu in the spaces provided. If you do not enter a file mode, the default is “A”.

If you want to select the files from a list, you can type an asterisk (*) for file name, file type, and/or file mode. An asterisk means that you want the list to contain *all* file names (or file types, or file modes).

You can also use two special characters in the file name and/or file type to request that the list contain a specific subset of files. The special characters are * (asterisk) and % (percent), where:

***** represents any *number* of character(s). As many asterisks as required can appear *anywhere* in a file name or file type.

% means any *single* character, but any character will do. As many percent symbols as necessary may appear anywhere in a file name or file type.

To display the list, first finish filling out the menu, and then press PF5. A special FILELIST screen is displayed instead of the SENDFILE menu. You select the files by typing a letter “s” in front of the file name of each file to be sent. Then press the ENTER key to send the files.

Another way to select files to be sent from the FILELIST screen is to position the cursor on the line describing a file you want to send, and then press PF5.

The Recipient(s):

You type the name(s) of the recipient(s) in the space provided. A name can take any of the forms listed above, in the “name” operand description.

The Options:

A list of options also appears on the menu. The default for each option appears to its left. You type 1 for YES or 0 for NO over any options for which you do not want the default. The options are as follows:

0 Request acknowledgment when the file has been received?

Type 1 for YES only if you want to get an acknowledgment when the person receives your file. The acknowledgment shows the date and time the file was received, and the recipient's user ID and node.

When you get an acknowledgment, it appears in your reader. If you choose to receive it, an entry is made in a "userid NETLOG" file, which is explained below.

1 Make a log entry when the file has been sent?

Each time you send a file, an entry is automatically made in the file "userid NETLOG." A typical entry might look like this:

File MY DATA A1 sent to JONES at NODE1 on 10/10/81 11:30:25 EDT

If you specified 1 on the first option (acknowledgment), an entry is also made when you receive the acknowledgment.

Type 0 if you don't want an entry made in the log file.

1 Display the file name when the file has been sent?

The names of the file(s) and the user ID(s) and node(s) of the recipients are displayed on a cleared screen. Type 0 if you do not want this information displayed.

0 This file is actually a list of files to be sent?

See the FILELIST command, the usage note "Saving a List of Files," for information on saving a file list. By saving a list of files created by either the FILELIST command or the LISTFILE command issued with the EXEC option, you can send all the files (and the list of files) at once.

Type 1 if your file is a list of files.

Sending a File:

If you specified only one file ID, press either PF5 or the ENTER key after filling out the SENDFILE menu. PF5 sends the file and exits from the menu. Pressing the ENTER key sends the file but keeps the menu.

If you are selecting files from a FILELIST screen type a letter "s" in front of each file name you want to send. Then press the ENTER key to send the file(s).

Keys on the SENDFILE Menu:

ENTER	Execute the command typed on the command line, or if none, send the file. (The ENTER key is set by the XEDIT subcommand, SET ENTER IGNORE MACRO EXECUTE.)
PF 1 Help	Display information about the SENDFILE command.
PF 2	Not assigned.
PF 3 Quit	Exit from the menu.
PF 4	Not assigned
PF 5 Send	Send the file(s) and exit from the menu.
PF 6	Not assigned
PF 7	Not assigned
PF 8	Not assigned
PF 9	Not assigned
PF 10	Not assigned

- PF 11** Not assigned
- PF 12 Cursor** If cursor is on the menu, move it to the command line; if cursor is on the command line, move it back to its previous location on the menu.

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

On a display terminal without PF keys, you can enter QUIT from the command line to exit from the screen.

Pressing the PA1 key while in the SENDFILE menu displays the WM window, unless the CP TERMINAL BRKKEY has been assigned to PA1.

Keys on the FILELIST Screen:

- ENTER** Execute the command(s) typed on file line(s), or on the command line. (The ENTER key is set by the XEDIT subcommand, SET ENTER IGNORE MACRO EXECUTE.)
- PF 1 Help** Display information about the FILELIST command.
- PF 2 Refresh** Update the list to indicate new files, erased files, etc., using the same parameters as those specified on the SENDFILE panel.
- PF 3 Quit** Exit from the list.
- PF 4 Sort** files by file type, file name.
- PF 5 Sendfile** at cursor. Append the fn ft fm on this line and send the file.
- PF 6 Sort** files by size, largest first.
- PF 7 Backward** Scroll backward one screen.
- PF 8 Forward** Scroll forward one screen.
- PF 9 FL/n** Issue the command FILELIST /n * * at the cursor, so that a list is displayed, containing all files that have the file name that is displayed on the line with the cursor.
- PF 10** Not assigned.
- PF 11 XEDIT** Edit the file pointed to by the cursor.
- PF 12 Cursor** If cursor is in the file area, move it to the command line; if cursor is on the command line, move it back to its previous location in the file.

In addition to setting the above PF keys, the PROFSEND XEDIT macro sets synonyms that you can use to sort your FILELIST files. Enter the synonyms on the SENDFILE command line. The synonyms are:

- SNAME** Sorts the list alphabetically by file name, file type, and file mode.
- STYPE** Sorts the list alphabetically by file type, file name, and file mode.
- SMODE** Sorts the list by file mode, file name, and file type.
- SRECF** Sorts the list by record format, file name, file type, and file mode.
- SLREC** Sorts the list by logical record length and then by size (greatest to least).
- SSIZE** Sorts the list by number of blocks and number of records (greatest to least).
- SDATE** Sorts the list by year, month, day, and time (most recent to oldest).

An example of a SENDFILE menu and a FILELIST screen are shown in the "Examples" section, below.

3. Format of the File Sent by SENDFILE

The format of the file that is sent depends on whether the OLD or NEW (the default) option is specified.

Important note: Unless the OLD option is specified, the RECEIVE command is the *only* way you can read a file sent by SENDFILE.

The OLD Option:

If the OLD and NOTE options are specified and the width (LRECL) of the note (prepared using the NOTE command) is 80 or less, SENDFILE uses the PUNCH command (with the HEADER option) to send the file. Otherwise, DISK DUMP is used to send the file. The OLD option should be used if the recipient does not have the RECEIVE command available to read the file.

The NEW Option:

If the NEW option is specified, control records are added and the file is sent in a format called "NETDATA." The NETDATA format is described in the *VM/SP CMS Diagnosis Reference*.

The transmitted file is composed of several control records, followed by the data records, and ending with a trailer record. If the file is an acknowledgment, it consists only of control records. An acknowledgment can be requested only with the NEW option.

The NEW option should be used when the recipient can read the file with the RECEIVE command on his CMS system, or when the file is being sent to the MVS operating system with TSO Extensions program product.

4. Priority

When SENDFILE is issued with the NEW option to send a file across the network (to a node different from yours), the file is assigned a priority. The order and speed of transmission are based on both this priority and the size of the file.

The priorities are assigned as follows:

NOTE files at least ten blocks in size: Priority = 00 (high)

Other files: Priority = 50 (medium)

Acknowledgments: Priority = 90 (low)

5. SENDFILE always sends files as CLASS A NOCONT NOHOLD regardless of the class to which you spool your PUNCH. The CP message that is generated, containing the spool ID, etc., is suppressed.
6. If you want to issue SENDFILE from an exec program, you should precede it with the EXEC command; that is, specify
exec sendfile

Examples

The following is a sample SENDFILE menu:

```

----- SENDFILE -----
File(s) to be sent (use * for Filename, Filetype and/or Filemode
                    to select from a list of files)
Enter filename : *
               filetype : data
               filemode : a

Send files to : sleepy

Type over 1 for YES or 0 for NO to change the options:

0 Request acknowledgement when the file has been received?
1 Make a log entry when the file has been sent?
1 Display the file name when the file has been sent?
0 This file is actually a list of files to be sent?

1= Help          3= Quit          5= Send          12= Cursor

====>
Macro-read 1 File

```

Figure 20. Sample SENDFILE Menu

In Figure 20, the sender typed an asterisk for file name, “data” for file type, and “a” for file mode. The name of the recipient (sleepy) is also typed on the screen. When PF5 is pressed, a special FILELIST screen is displayed. The files to be sent can be selected from this screen (shown in Figure 21).

```

SNOWHITE FILELIST A0 V 108 Trunc=108 Size=418 Line=1 Col=1 Alt=0
Cmd  Filename Filetype Fm Format Lrec|  Records  Blocks  Date    Time
s    WISTFUL  DATA   A1 V    95      34       2 10/04/80 21:12:04
s    BOSS    DATA   A1 V    95      29       2 10/04/80 20:58:07
    DUMMY    DATA   A1 V   107     281      10 10/04/80 17:59:00
s    GROUCHY DATA   A1 V    92     101       4 10/02/80 15:33:05
    PRINCE  DATA   A2 V    75      28       1  9/25/80 12:10:03
s    SNOOZY  DATA   A2 V   120     277      10  9/24/80  9:14:02
    SNIFFLES DATA   A1 V    26       7       1  9/23/80 16:50:06
    WITCH   DATA   A1 V    80     489      30  8/26/80 16:05:08

1= Help    2= Refresh 3= Quit   4= Sort(type) 5= Sendfile 6= Sort(size)
7= Backward 8= Forward 9= FL /n 10=          11= XEDIT  12= Cursor
Type 'S' in front of each file to be sent, and press ENTER
====>
X E D I T 1 File

```

Figure 21. Sample FILELIST Screen Invoked from SENDFILE

To send one or more of these files, you can type a letter “s” in front of the file name of each file you want sent (see above) and then press the ENTER key. You can also

SENDFILE

position the cursor on the line describing the file you want to send, and then press the PF5 key.

Responses

Body of the note kept in 'fn NOTEBOOK fm'

Header only added to other NOTEBOOK files.

File|Note 'fn ft fm' sent to 'userid' at 'node' on 'date time timezone'

nnn files have been sent.

File *fn ft fm* not found.

Note added to *fn* NOTEBOOK *fm*

The following message appears on the FILELIST screen invoked from a SENDFILE menu:

Type 'S' in front of each file to be sent, and press ENTER.

Messages and Return Codes

DMSWSF002E File *fn ft fm* not found [RC = 28]
DMSWSF006E No read/write filemode accessed [RC = 36]
DMSWSF048E Invalid mode *mode* [RC = 24]
DMSWSF054E Incomplete fileid specified [RC = 24]
DMSWSF062E Invalid character * in fileid *fn ft fm* [RC = 20]
DMSWSF069E Filemode *mode* not accessed [RC = 36]
DMSWSF399E Too many tags or tag too long for *nickname* in *userid* NAMES file [RC = 88]

DMSWSF579E Records truncated to *nn* when added to *fn ft fm* [RC = 3]
DMSWSF637E Missing nodeid for the AT operand [RC = 24]
DMSWSF647E Userid not specified for *nickname* in *userid* NAMES file [RC = 32]
DMSWSF648E Userid *name* not found; no files have been sent [RC = 32]
DMSWSF667E NOTE header does not contain the {keyword From:|keyword To:|OPTIONS line|DATE line} [RC = 32]

DMSWSF671E Error sending file *fn ft fm*; rc = *nn* from *command* [RC = 100]
DMSWSF672E Virtual punch invalid or not defined [RC = 36]
DMSWSF673E Addressees are in the note header cards; do not specify names with NOTE option [RC = 24]

DMSWSF674E Punch is not ready [RC = 36]
DMSWSF675E No names specified [RC = 24]
DMSWSF676E Invalid character * for Network ID [RC = 20]
DMSWSF677E Invalid option: *option* in option line [RC = 32]
DMSWSF678E Invalid note header format; note cannot be sent [RC = 32]
DMSWSF679E Filemode *mode* is not accessed; note cannot be sent [RC = 36]
DMSWSF680E Invalid fileid specified with FILELIST option [RC = 20]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

Messages from the SENDFILE Panel

DMSWSF002E File *fn ft fm* not found [RC = 28]
 DMSWSF048E Invalid mode *mode* [RC = 24]
 DMSWSF054E Incomplete fileid specified [RC = 24]
 DMSWSF069E Filemode *mode* not accessed [RC = 36]
 DMSWSF081E Invalid reply; answer 1 for YES and 0 for NO
 DMSWSF637E Missing nodeid for the AT operand [RC = 24]
 DMSWSF647E Userid not specified for *nickname* in *userid* NAMES file [RC = 32]
 DMSWSF648E Userid *name* not found; no files have been sent [RC = 32]
 DMSWSF657E Undefined PFkey/PAkey
 DMSWSF675E No names specified [RC = 24]
 DMSWSF676E Invalid character * for Network ID [RC = 20]
 DMSWSF680E Invalid fileid specified with FILELIST option [RC = 20]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813

SENTRIES

Use the SENTRIES command to determine the number of lines currently in the program stack. When you issue a SENTRIES command, CMS returns the number of lines in the program stack (but not the terminal input buffer) as a return code.

Format

SENTRIES	
----------	--

Usage Note

If you issue a SENTRIES command in an exec that has set up a procedure to be done when an error occurs, a nonzero SENTRIES return code causes that procedure to execute.

SET

Use the SET command to establish, turn off, or reset a particular function in your CMS virtual machine. Only one function may be specified per SET command. SET cannot be issued without an option.

The options available with SET are summarized below. A complete description of each option follows.

For descriptions of SET commands that pertain to using windows, see "SET" on page 727.

ABBREV	IMESCAPE	PROTECT
APL	IMPCP	REDTYPE
AUTOREAD	IMPEX	RDYMSG
BLIP	INPUT	RELPAGE
CMSTYPE	INSTSEG	REMOTE
COMDIR	KEYPROTECT	SERVER
DOS	LANGUAGE	STORECLR
DOSLNCNT	LDRTBLS	SYSNAME
DOSPART	LINEND	TEXT
EXECTRAC	LOADAREA	THReshold
FILEPool	NONDISP	TRANslate
FILEWait	NONSHARE	UPSI
FULLREAD	OUTPUT	

Usage Notes

1. If you issue the SET command specifying an invalid function and the implied CP function is in effect, you may receive message DMKCFC003E Invalid option - *option*
2. If an invalid SET command function is specified from an exec and the implied CP function is in effect, then the return code is -0003.
3. To determine or verify the setting of most functions, use the QUERY command.

Messages and Return Codes

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

SET ABBREV

Use the ABBREV option to control whether the system accepts system and user abbreviations for command names and their translations, or only full command names or the full synonym or translation.

Format

SET	ABBREV { ON OFF }
------------	---

Operands

ON

accepts system and user abbreviations for command names and their translations. The SYNONYM command makes synonym abbreviations available. The SET TRANSLAT command makes translation abbreviations available.

For example, if GETDISK is a synonym for ACCESS and the minimum abbreviation is three, then you can enter GET, GETD, GETDI, GETDIS, or GETDISK to issue the ACCESS command. The same is true if GETDISK is a translation of ACCESS.

OFF

accepts only the full command name or the full synonym or translation (if one is available) for the command name.

For a discussion of the relationship of the SET ABBREV and SYNONYM commands, refer to the SYNONYM command description.

Initial Setting

ABBREV ON

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET APL

Use the APL option to activate character code conversion for APL characters for the System Product Editor and CMS.

Format

SET	APL { ON OFF }
------------	--

Operands

ON

activates character code conversion for APL characters. Before using APL keys, issue SET APL ON to ensure proper character code conversion.

OFF

specifies that no character code conversion is performed for APL characters and keys.

Initial Setting

APL OFF

Usage Notes

1. The APL setting is valid only when performing full-screen I/O (for example, in XEDIT or in CMS with SET FULLSCREEN ON). If you are in CP or using a line-mode terminal, SET APL has no effect.
If you are in CP, you can issue the TERMINAL APL ON command to have CP convert APL character codes.
2. Because the APL character code conversion is costly, it is recommended that you issue SET APL OFF when you stop using the special APL keys.
3. When SET APL ON is specified, TEXT is set OFF.
4. Changing the APL setting for CMS also changes the APL setting for the System Product Editor, and vice versa.

Messages and Return Codes

DMSSEF109S	Virtual storage capacity exceeded [RC=104]
DMSSEF524W	NONDISP character reset to ".
DMSSEF926E	Command is only valid on a display terminal [RC=88]
DMSSET109S	Virtual storage capacity exceeded [RC=104]
DMSWIR329W	Warning: APL/TEXT option not in effect

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET AUTOREAD

Use the AUTOREAD option to specify whether or not a console read is to be issued after command execution.

Format

SET	AUTOREAD { ON } { OFF }
-----	----------------------------

Operands

ON

specifies that a console read is to be issued immediately after command execution. ON is the initial setting for non-display, non-buffered terminals.

OFF

specifies that you do not want a console read to be issued until you press the Enter key or its equivalent. OFF is the initial setting for display terminals because the display terminal does not lock, even when there is no READ active for it.

Note: If you disconnect from one type of terminal and reconnect on another type, the AUTOREAD status remains unchanged.

Initial Setting

AUTOREAD ON for non-display, non-buffered terminals.

AUTOREAD OFF for display terminals.

Usage Note

Your virtual machine may be logged on automatically if it processes private resource connection requests. If your virtual machine processes private resource connection requests, put the statement SET AUTOREAD OFF in your PROFILE EXEC to allow the processing of private resource connection requests.

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET BLIP

CMS does not support the BLIP facility. If you issue the SET BLIP command, CMS ignores the operand and the BLIP setting remains OFF. The command is maintained for compatibility only.

SET CMSTYPE

Use the CMSTYPE option to suppress or resume CMS terminal display within an exec.

Format

SET	CMSTYPE { HT RT }
-----	----------------------

Operands

HT

suppresses CMS terminal display within an exec. All CMS terminal display from an exec, except for CMS error messages with a suffix letter of 'S' or 'T', is suppressed until the end of the exec file or until a SET CMSTYPE RT command is executed. Some CMS commands may reset CMSTYPE to RT. In general, those commands that interact with the user through the console (i.e. HELP, XEDIT, or any command or module that issues a READ to the console or the &READ EXEC control word) may reset CMSTYPE.

RT

resumes CMS terminal display which has been suppressed as a result of a previous SET CMSTYPE HT command.

Initial Setting

CMSTYPE RT

Usage Notes

1. &STACK HT and SET CMSTYPE HT have the same effect when interpreted by the CMS EXEC processor. Similarly, &STACK RT and SET CMSTYPE RT are equivalent for the CMS EXEC processor. However, when using EXEC 2, the commands &STACK HT and &STACK RT cause the characters "HT" and "RT" to be placed in the program stack and do not affect the console output. These characters must be used by a program or cleared from the stack. Otherwise, you will receive an "UNKNOWN CP/CMS COMMAND" error message when they are read from the program stack.
2. In fullscreen-CMS, SET CMSTYPE HT purges nonpriority output that is in the queue for the virtual screen to which message class CMS is routed.

Messages and Return Codes

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET COMDIR

Use the COMDIR option to set up and control the CMS communications directory. For more information on the CMS communications directory, see the *VM/SP Connectivity Planning, Administration, and Operation*.

Format

SET	COMDIR	FILE	{ SYSTEM USER }	{ NONE <i>fileid</i> }
		ON	[SYSTEM USER BOTH]	}
		OFF	[SYSTEM USER BOTH]	
		RELOAD	[SYSTEM USER BOTH]	

Operands

FILE

sets the file name of the directory.

ON

turns on name resolution.

OFF

turns off name resolution.

RELOAD

causes the static memory image of the name resolution data base to be reloaded.

SYSTEM

USER

BOTH

indicates the directory level to set. The BOTH option is the default for the ON, OFF, and RELOAD operands.

NONE

indicates that you do not want a file associated with the specified directory level. If a directory is already loaded for the specified directory level, it is unloaded.

SET COMDIR

fileid

specifies the file to associate with the specified directory level and loads the file into memory. The file ID is expressed as *fn ft fm*.

Messages and Codes

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

SET DOS

Use the DOS option to place your virtual machine in the CMS/DOS environment or return it to the normal CMS environment. In addition, you can specify:

- The file mode letter at which the VSE system residence is accessed.
- That you are going to use the AMSERV command or you are going to execute programs to access VSAM data sets.

Format

SET	DOS { ON [<i>fm</i> [(VSAM [])]]] } OFF
------------	---

Operands

ON

places your CMS virtual machine in the CMS/DOS environment. The logical unit SYSLOG is assigned to your terminal.

fm

specifies the file mode letter at which the VSE system residence is accessed; the logical assignment of SYSRES is made for the indicated file mode letter.

CMS/DOS support is based on the VSE/AF 1.3.5 program product and does not support the VSE/AF 2.1 libraries.

VSAM

specifies that you are going to use the AMSERV command or you are going to execute programs to access VSAM data sets.

OFF

returns your virtual machine to the normal CMS environment. All previously assigned system and programmer logical units are unassigned.

Initial Setting

DOS OFF

Messages and Return Codes

DMSSET048E	Invalid mode <i>mode</i> [RC = 24]
DMSSET109S	Virtual storage capacity exceeded [RC = 104]
DMSSET402W	DMSLBR no in CMSBAM segment; ESERY support not available.
DMSSET410S	Control program error indication <i>xxx</i> [RC = <i>rc</i>]
DMSSET444E	Volume <i>valid</i> is not a DOS SYSRES [RC = 32]
DMSSET804E	Error establishing CMS/DOS environment [RC = <i>nm</i>]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET DOS

Note: In $RC=rc$, the rc represents the actual error code generated by CP.

SET DOSLNCNT

Use the DOSLNCNT option to specify the number of SYSLST lines per page for the CMS/DOS environment.

Format

SET	DOSLNCNT <i>nn</i>
------------	---------------------------

Operands

DOSLNCNT *nn*

specifies the number of SYSLST lines per page. *nn* is an integer from 30 to 99.

Initial Setting

056

Messages and Return Codes

DMSSET099E CMS/DOS environment not active [RC = 40]

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET DOSPART

Use the DOSPART option to specify the size of the partition in which you want a program to execute in the CMS/DOS environment.

Format

SET	DOSPART { <i>nnnnnK</i> OFF }
------------	--

Operands

nnnnnK

specifies the size of the virtual partition in which you want a program to execute. The value, *nnnnnK*, may not exceed the amount of user free storage available in your virtual machine.

You should use this function only when you can control the performance of a particular program by reducing the amount of available virtual storage.

Note: In rare circumstances, it may happen that when a program is executed, the amount of storage available is less than the current DOSPART. Then, only the amount of storage available is obtained; no message is issued.

OFF

specifies that you no longer want to control your virtual machine partition size. When the DOSPART setting is OFF, CMS computes the partition size whenever a program is executed.

Initial Setting

DOSPART OFF

Usage Notes

1. When a DOS partition is defined, message DMSSET1101I is issued to tell you how much space was obtained and where the partition is located. If DOS cannot get the requested amount of storage, then it gets as much as it can and tells you how much it obtained.

It is not an error if DOS obtains less than the requested amount of storage.

2. Error message DMSSET333E is given only if DOS is unable to get the minimum amount of storage it needs. See the CP DEFINE command in the *VM/SP CP General User Command Reference* for information on getting more storage.

Messages and Return Codes

DMSSET099E	CMS/DOS environment not active [RC = 40]
DMSSET109S	Virtual storage capacity exceeded [RC = 104]
DMSSET333E	<i>nnnnnK</i> partition too large for this virtual machine [RC = 24]
DMSSET1101I	<i>nnnnnK</i> DOS partition defined at hexadecimal location <i>xxxxxx</i> [RC = 0]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

	Reason	Page
)	Errors in command syntax	811

SET EXECTRAC

Use the EXECTRAC option set tracing on or off for your System Product Interpreter or EXEC 2 program.

Format

SET	EXECTRAC { ON OFF }
------------	--------------------------------

Operands

ON

specifies that you want tracing turned on for your System Product Interpreter or EXEC 2 program. The tracing bit in the EXECFLAG in NUCON is turned on. The tracing bit is cleared upon return to CMS or XEDIT, turning the tracing off.

OFF

specifies that you want tracing turned off for your System Product Interpreter or EXEC 2 program.

Initial Setting

EXECTRAC OFF

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET FILEPOOL

Use the FILEPOOL option to set (or reset) your default file pool. The default file pool ID will be used when you do not specify a specific file pool ID in a command or function call. This default lasts until the CMS session ends.

To determine what file pool is currently set as the default, use the QUERY FILEPOOL CURRENT command. To see what the default file pool was initially the default, use the QUERY FILEPOOL PRIMARY command.

Format

SET	FILEPool [<i>filepoolid:</i> NOne PRImary]
------------	---

Operands

filepoolid:

is the name of the file pool that you want as the default file pool. Remember that you must always follow the filepoolid with a colon.

NOne

specifies that you do not want a default file pool.

PRImary

resets the default file pool to the setting when you began your CMS session. PRIMARY is the default.

Usage Notes

1. The default file pool at IPL can be defined by:
 - including the FILEPOOL keyword in the IPL statement in your CP directory entry. This is done by your system administrator.
 - using the FILEPOOL keyword in your IPL statement, for example:
ipl cms parm filepool vmsysu
 - entering the SET FILEPOOL command from the command line.
 - if your 191 disk is accessed as file mode A, placing SET FILEPOOL *filepoolid:* in your PROFILE EXEC.
2. If you issue the ACCESS command with no operands and you have a default file pool, the top directory in your default file pool is accessed as your file mode A. If you do not have a default file pool, your 191 disk is accessed as A.

SET FILEPOOL

Messages and Return Codes

DMSJSE109S Virtual storage capacity exceeded [RC = 104]
DMSJSE389E Invalid operand: *operand* [RC = 24]
DMSJSE391E Unexpected operand(s): *operand* [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET FILEWAIT

Use the FILEWAIT option to indicate whether or not you want a request for control of an SFS directory or a file in a directory to wait until it becomes available.

Format

SET	FILEWai t { ON } { OFF }
------------	--

Operands

ON

indicates that you do want to wait until the requested file or directory becomes available. The wait may be for a prolonged period of time.

OFF

indicates that you do not want to wait until the requested file becomes available. The request fails immediately if the file is not available.

Initial Setting

FILEWAIT OFF

Usage Notes

1. The FILEWAIT setting is the same for all your file pools.
2. Even with FILEWAIT ON, a request will not wait for a file or directory that was locked by the CREATE LOCK command.
3. With FILEWAIT ON, if you request an explicit lock with the CREATE LOCK command and the file or directory has an implicit lock on it, your request will wait until the implicit lock is deleted.
4. With FILEWAIT ON, if you want to write to a file that another user has open to write, your request will wait until the implicit lock is deleted.
5. A request may not wait even though you have set FILEWAIT ON, to avoid a deadlock situation. For example, UserA holds a lock on file #1 while waiting for a lock on file #2, while UserB holds a lock on file #2 while waiting for a lock on file #1.

Example

You have a program that updates a file. You issue:

```
set filewait on
```

When your program tries to update the file and another user has it open to write to it, your request will wait until the user closes the file. If the user has also locked the file, using the CREATE LOCK command, your request will not wait.

SET FILEWAIT

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET FULLREAD

Use the FULLREAD option to allow 3270 null characters to be recognized in the middle of the physical screen by CMS and the System Product Editor.

Format

SET	FULLREAD { ON } { OFF }
------------	---

Operands

ON

enables nulls to be recognized in the middle of lines, making it easier for you to enter tabular or pictorial data.

OFF

inhibits transmission of nulls from the terminal.

Initial Setting

OFF

Usage Notes

1. When FULLREAD ON is issued, nulls at the end of screen lines that are part of a logical line that occupies more than one physical screen line are dropped. This allows you to delete characters in a screen line and still have the line reconstructed flush together even though multi-line 327X lines do not “wrap” when the character delete key (or the insert mode key) is used.
2. FULLREAD ON increases communication to the CPU, which generally results in increased response time.
3. Setting FULLREAD ON will prevent you from losing any screen changes when you press a PA key and a message is displayed on a cleared screen.
4. A certain terminal configuration, which imposes several restrictions on your session, occurs when going through a VM/Passthru Facility (5749-RC1) (PVM) 327X Emulator link to another VM system. These PVM links can be identified by an S to the immediate left of the node ID in the PVM selection screen. The following is a list of these restrictions:
 - a. The SET FULLREAD ON command may not be used.
 - b. All PA keys (except for the CP defined TERMINAL BRKKEY) are non-functional.
5. Changing the FULLREAD setting for CMS also changes the FULLREAD setting for the System Product Editor.

SET FULLREAD

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
DMSSEF926E Command is only valid on a display terminal [RC = 88]
DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET IMESCAPE

Use the IMESCAPE to indicate whether or not an escape character is required to execute immediate commands.

Format

SET	IMESCAPE { ON OFF <i>char</i> }
------------	--

Operands

ON

indicates that an escape character is required to execute immediate commands. The default escape character is a semi-colon (;).

OFF

indicates that an escape character is not required to execute immediate commands. This is the default setting.

char

indicates that an escape character is required to execute immediate commands. The escape character is a single character and it cannot be A-Z or 0-9.

Initial Setting

IMESCAPE OFF

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET IMPCP

Use the IMPCP option to control handling of command names that CMS does not recognize. Unknown commands may be considered CP commands or an error.

Format

SET	IMPCP { ON } { OFF }
-----	-------------------------

Operands

ON

passes command names that CMS does not recognize to CP; that is, unknown commands are considered to be CP commands.

OFF

generates an error message at the terminal if a command is not recognized by CMS.

Initial Setting

IMPCP ON

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET IMPEX

Use the IMPEX option to control whether or not exec files are treated as commands.

Format

SET	IMPEX { ON } { OFF }
------------	--

Operands

ON

treats exec files as commands; an exec file is invoked when the file name of the exec file is entered.

OFF

does not consider exec files as commands. You must issue the EXEC command to execute an exec file.

If you issue SET IMPEX OFF, you may not be able to use PF keys predefined to perform exec functions in productivity aids such as NOTE, RDRLIST, FILELIST, etc.

Initial Setting

IMPEX ON

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET INPUT

Use the INPUT option to translate characters entered from your terminal to hexadecimal code. You can also reset the hexadecimal code to a specified hexadecimal code in your translate table.

Format

SET	INPUT $\left[\begin{array}{l} a \ xx \\ xx \ yy \end{array} \right]$
------------	--

Operands

INPUT a xx

translates the specified character a to the specified hexadecimal code xx for characters entered from the terminal.

INPUT xx yy

allows you to reset the hexadecimal code xx to the specified hexadecimal code yy in your translate table.

Note: If you issue SET INPUT and SET OUTPUT commands for the same characters, issue the SET OUTPUT command first.

INPUT

returns all characters to their default translation.

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET INSTSEG

Use the INSTSEG option to specify whether or not the system should search the CMS installation saved segment to locate an exec or System Product Editor macro.

Note: Execs in loaded saved segments, that had INSTSEG specified when the segment was built, are considered logically as part of the CMS installation saved segment and are affected by this command.

Format

SET	INSTSEG { ON [<i>fm</i> LAST] } OFF
------------	--

Operands

ON

indicates that you want the system to search the CMS installation saved segment when locating an exec. The initial setting is ON.

fm

LAST

specifies the location of the CMS installation saved segment in the command search order. It is searched immediately before the disk or directory having the file mode letter that you specify. If you haven't accessed a disk or directory as that file mode, the CMS installation saved segment will be searched in that position. The default file mode is S. LAST indicates that the saved segment will be searched after all accessed disks and directories have been searched.

OFF

indicates that you do not want the system to search the CMS installation saved segment when locating an exec.

Initial Setting

INSTSEG ON

Messages and Return Codes

DMSSET048E Invalid mode *mode* [RC=24]

DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET KEYPROTECT

Use the SET KEYPROTECT command to reset the user keys, X'E0', when a DMSFRET or CMSSTOR RELEASE occurs.

Format

SET	KEYPROTECT { ON OFF }
------------	---

Operands

ON

resets the storage keys for the whole virtual machine if the previous setting was *OFF*, according to the values defined by storage management. It does not reset the nonshared pages.

The user keys are reset whenever a DMSFRET or CMSSTOR RELEASE occurs. If an ABEND occurs, the storage keys for all pages allocated in the following areas will be reset when storage returns:

- User subpool
- Private subpools
- Shared subpools
- Nonsystem global subpools in user key.

OFF

does not reset the user keys when a DMSFRET or CMSSTOR RELEASE occurs.

Initial Setting

OFF

Usage Notes

1. If user programs set keys, they must restore the keys to their original settings. If there are programs that depend on CMS resetting user keys, issue SET KEYPROTECT ON to insure that the user keys are set properly.
2. To check the setting of KEYPROTECT, issue:

QUERY KEYPROTECT

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET LANGUAGE

Use the SET LANGUAGE command to change the current language of your CMS session and any application running on CMS that uses national language support. When you SET another language, you will be able to receive messages, enter commands, and see CMS panels (like the FILELIST screen) in that language.

The SET LANGUAGE command also informs CP to change the language used to issue CP messages to your virtual machine.

Format

SET	LANGUAGE [<i>langid</i>] [(options ... [])] <u>Options:</u> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 2px;"> ADD <i>applid</i> DELETE <i>applid</i> </div> <div style="border: 1px solid black; padding: 2px;"> USER SYSTEM ALL </div> <div style="border: 1px solid black; padding: 2px;"> TYPE NOTYPE </div> </div>
------------	---

Operands

langid

is the language identifier of the language to which the virtual machine will be set. A language ID may be one to five characters in length and must be made up of only CMS file system characters. When you specify *langid*, SET LANGUAGE checks VMFNLS LANGLIST, and gets the one or two-character country code for the language you specified. If you do not specify *langid*, then SET LANGUAGE gets the one or two-character country code from the VMFNLS LANGLIST for the current language.

Options

ADD

activates the system and/or user language files, for the application named (*applid*). (You can make additions to system message repositories and the CMS command syntax file; see the *VM/SP Application Development Guide for CMS* for details.)

If you specify ADD and a language ID that is different from the current language setting, then

- the current language is changed for all active applications,
- the application named is activated, and
- a request is made for CP to change the language used to issue CP messages to your virtual machine.

DELETE

deactivates the system and/or user language files, for the application named.

applid

specifies the application whose system and/or user language files that should be added or deleted. The application ID must be three characters in length.

SET LANGUAGE

USER

specifies that user repositories, and/or command syntax tables, and/or command synonym tables are loaded into storage or removed from storage. See Usage Note 1.

SYSTEM

specifies that the saved segment with system-provided language files for the application named is activated or deactivated. No user language files are affected if you specify SYSTEM. This is the default.

ALL

specifies that the saved segment with system-provided language files *and* user additions are activated or deactivated.

TYPE

specifies that all messages from the SET LANGUAGE command are to be typed to the console. This is the default.

NOTYPE

specifies that no messages from the SET LANGUAGE command are to be typed to the console. Messages resulting from syntax errors will be displayed even if NOTYPE is specified.

Usage Notes

1. SET LANGUAGE searches for the file name(s) that have a file type of TEXT. When a file type of TEXT is found, SET LANGUAGE loads that file. If a file type of TEXT is not found, SET LANGUAGE looks for a file type of *TXLangid*, and loads it if found. For the USER option, CMS searches for the following file IDs with a file type of TEXT:

File ID	Description
<i>applidUMecc</i> TEXT	User message repositories.
<i>applidUPAcc</i> TEXT	User command syntax definitions.
<i>applidUSYcc</i> TEXT	User national language translation and synonyms. This file is generated automatically from user additions to the Definition Language for Command Syntax (DLCS) file. For more information about DLCS, see the <i>VM/SP Application Development Guide for CMS</i> .

where

applid

is the three-character application identifier.

cc

is the one or two-character country code that is appended to the file names of the object files for the current national language or the language you have specified.

2. The QUERY LANGLIST command displays all the valid language IDs that you can set in CMS. (Note that to set a language in CMS, the CP language files must also be available.) To display the language that is active for CMS in your virtual machine, issue the QUERY LANGUAGE command. Contact your system administrator if you have any questions about the languages available on your VM/SP system.
3. To delete the application DMS, use the USER option—you cannot delete DMS with the SYSTEM or ALL options.

4. If you ADD an application that already has active language files, your new language files for that application will replace the current language files.
5. Be aware of language-related terminal restrictions when specifying a language. If multiple CMS languages are to be available, then multiple language configurations need to be available for the hardware. This will insure that special characters for each language are displayed properly on the hardware. Certain EBCDIC character representations can represent different characters, depending on which language is defined in the hardware. For example, a X'D0' represents one character in English, and another character in French. If the software language does not match the language defined in the hardware, the data displayed at the terminal may not be what was expected.
6. When you issue the SET LANGUAGE command in full-screen CMS, the reserved lines are not updated to reflect the new language. To update the reserved lines, issue SET FULLSCREEN OFF, enter the SET LANGUAGE command, and then issue SET FULLSCREEN ON.

Similarly, if you are in a CMS environment such as FILELIST, MACLIST, RDRLIST, and SENDFILE, and you issue the SET LANGUAGE command, the reserved lines are not updated to reflect the new language. To update the reserved lines, exit the environment, enter the SET LANGUAGE command, and then issue the command to enter the environment.

7. If you only have object files with TXT*langid* as the file type and no file types of TEXT, then you must first access a read/write minidisk or SFS directory as A before issuing SET LANGUAGE with the ADD, USER, or ALL options. If all your object files have a file type of TEXT, then a read/write access is not necessary.
8. To avoid unpredictable results, you should issue SET LANGUAGE before you load programs that require user repositories, command syntax tables, or command synonym tables.

If the tables and repositories are also used by other CMS applications, then place them in shared storage. For information on how to do this, see the sections about National Language files in the *VM/SP Application Development Guide for CMS*.

Example

Suppose your virtual machine is set to American English (*langid*=AMENG) and you are working in CMS (*applid*=DMS).

- To work in French (*langid*=FRANC) and run an application called APPLICATION1 (*applid*=AP1), enter:

```
set language franc ( add ap1 system
```

which changes the language to French for CMS and APPLICATION1, and informs CP to change its language to French.

- To return to American English, enter:

```
set language ameng
```

which changes CMS and APPLICATION1 to American English, and tells CP to change its language back to American English (if APPLICATION1 is available in American English).

- To deactivate APPLICATION1, yet preserve American English as the current language, you enter:

SET LANGUAGE

set language (delete apl all

which returns your virtual machine to where you started: running CMS in American English with no other active applications.

Messages and Return Codes

DMSSSET109S	Virtual storage capacity exceeded [RC = 104]
DMSSLG005E	No application id specified [RC = 24]
DMSSLG109S	Virtual storage capacity exceeded [RC = 104]
DMSSLG276E	Invalid language id <i>langid</i> [RC = 24]
DMSSLG277E	The saved segment is located partially or entirely inside the virtual machine [RC = 88]
DMSSLG278E	Unable to set requested language: <i>langid</i> . <i>langid</i> forced [by CP] [RC = 104]
DMSSLG278E	The requested language <i>langid</i> is not available; <i>langid</i> forced [by CP] [RC = 104]
DMSSLG279E	Application <i>applid</i> not found in the language saved segment [RC = 28]
DMSSLG279I	Application <i>applid</i> not found in the language saved segment
DMSSLG280E	Application <i>applid</i> not active [RC = 28]
DMSSLG281E	Application DMS cannot be deleted [RC = 24]
DMSSLG332E	No user additions were loaded [RC = 28]
DMSSLG332I	No user additions were loaded
DMSSLG334E	No system information or user additions were found for application <i>applid</i> [RC = 28]
DMSSLG346E	Error <i>nn</i> loading <i>fn ft fm</i> from disk or directory [RC = 32]
DMSSLG639E	Error in <i>name</i> routine; return code was <i>nn</i>
DMSSLG770E	Invalid application id <i>applid</i> [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

SET LDRTBLS

Use the LDRTBLS option to define the number of pages of storage to be used for loader tables.

Format

SET	LDRTBLS [<i>nn</i>]
-----	-----------------------

Operands

LDRTBLS *nn*

defines the number (*nn*) of pages of storage to be used for loader tables. To set the size of the loader tables, you may issue the SET LDRTBLS command anytime after IPL. By default, a virtual machine having up to 384K of addressable real storage has two pages of loader tables; a larger virtual machine has three pages. Each loader table page has a capacity of 204 external names.

During LOAD and INCLUDE command processing, each unique external name encountered in a TEXT deck is entered in the loader table. The LOAD command clears the table before reading TEXT files; INCLUDE does not. This number can be changed with the SET LDRTBLS *nn* command provided that:

- *nn* is a decimal number between 1 and 127 inclusive, and
- the virtual machine has enough storage available to allow *nn* pages to be used for loader tables.

If these two conditions are met, *nn* pages are set aside for loader tables.

If you plan to change the number of pages allocated for loader tables, you should de-allocate storage at the high end of storage so that the storage for the loader tables may be obtained from that area. Usually, you can de-allocate storage by releasing one or more of the disks that were accessed.

Initial Setting

Dependent on size of virtual machine.

Messages and Return Codes

DMSSET031E	Loader tables cannot be modified [RC = 40]
DMSSET109S	Virtual storage capacity exceeded [RC = 104]
DMSSET990I	Insufficient storage available to create the requested loader tables. The loader tables that existed when the SET LDRTBLS command was issued have been created. [RC = 0]
DMSSET991E	Insufficient storage available to create the loader tables [RC = 104]
DMSSET992I	Insufficient storage available to create the requested loader tables. The default loader tables have been created. [RC = 0]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET LINEND

SET LINEND

Use the LINEND option to activate and/or define the logical line end for full-screen CMS.

Format

SET	LINEND { ON OFF } [char]
-----	-----------------------------

Operands

ON

allows you to enter several commands on the same line, separated by the line end character.

OFF

specifies that the logical line end character is not recognized.

char

is the character to be used as a line end character. The default line end character is a pound sign (#).

Initial Setting

LINEND ON #

Usage Note

If you redefine the line end character to a symbol other than a pound sign (#), the #WM command is invalid. (Therefore, the default CMSPF keys that issue #WM commands do not function.) To use the #WM command you must substitute your line end symbol for the pound sign.

Messages and Return Codes

DMSSEF926E Command is only valid in CMS FULLSCREEN mode [RC=88]
DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET LOADAREA

Use the LOADAREA option to define the ORIGIN default for the load process. This SET command ONLY affects where TEXT files are to be loaded, and does not influence the RMODE that may be propagated to the GENMOD process.

The SET LOADAREA command behaves differently depending upon the environment it is issued from. In the following description a VM/SP virtual machine and a VM/XA SP System/370 mode virtual machine are equivalent; a VM/XA SP 370-XA mode virtual machine allows you to use addresses above the 16Mb line. The behavior of the SET LOADAREA command in both environments is described below to help you plan and develop applications that will run in both environments.

Format

SET	LOADAREA { 20000 RESPECT }
------------	--

Operands

20000

indicates that the LOAD command will start loading at X'20000' if ORIGIN or RMODE option is not specified on the LOAD command.

- If you specified ORIGIN, the load begins at the specified address.
- If you specified RMODE, the load begins at the largest contiguous area below 16Mb.

In an 370-XA mode virtual machine, the X'20000' setting overrides the RMODE definition encountered during TEXT ESD processing when the ORIGIN or RMODE option is not specified on the LOAD command.

RESPECT

indicates, in an 370-XA mode virtual machine, that the LOAD command will respect the RMODE during the TEXT ESD processing to determine where the loaded programs should reside when the RMODE or ORIGIN option is not specified on the LOAD command.

If you specified RMODE 24 or RMODE ANY on the first TEXT ESD, loading will begin at the largest contiguous area of storage allocated:

- Below 16Mb for RMODE 24
- Above 16Mb for RMODE ANY.

In a System/370 mode virtual machine, the SET LOADAREA RESPECT command will cause the LOAD command to start loading at the beginning of the largest contiguous area below 16Mb if the ORIGIN option is not specified on the LOAD command.

SET LOADAREA

Initial Setting

20000 in a System/370 mode virtual machine
 RESPECT in a 370-XA mode virtual machine

Examples

1. SET LOADAREA 20000 active

LOAD command	System/370 mode machine	370-XA mode machine
LOAD pgma ...	Load begins at 20000, RMODE/AMODE ignored	Load begins at 20000, overrides RMODE on TEXT(s) ESD, AMODE on TEXT ESD respected
LOAD pgma ..(RMODE 24	Load begins at the largest contiguous area under 16Mb, AMODE on TEXT ESD respected unless AMODE override on LOAD command	Load begins at the largest contiguous area under 16Mb, AMODE on TEXT ESD respected unless AMODE override on LOAD command
LOAD pgma ..(RMODE ANY	Load begins at the largest contiguous area under 16Mb, AMODE on TEXT ESD respected unless AMODE override on LOAD command	Load begins at the largest contiguous area above 16Mb, AMODE on TEXT ESD respected unless AMODE override on LOAD command
LOAD pgma ..(ORIGIN TR	Load begins at start of transient area, SET LOADAREA is overridden, AMODE ignored	Load begins at start of transient area, SET LOADAREA is overridden, AMODE on TEXT ESD respected unless AMODE override on LOAD command
LOAD pgma ..(ORIGIN <i>hexloc</i>	Load begins at area specified by hexloc, SET LOADAREA is overridden, AMODE ignored	Load begins at area specified by hexloc, SET LOADAREA is overridden, AMODE on TEXT ESD respected unless AMODE override on LOAD command

2. SET LOADAREA RESPECT active

LOAD command	System/370 mode machine	370-XA mode machine
LOAD pgma ...	Load begins at the largest contiguous area under 16Mb, RMODE/AMODE ignored	Load begins at the largest contiguous area according to RMODE determined from TEXT(s) ESD, AMODE on TEXT ESD respected
LOAD pgma ..(RMODE 24	Load begins at the largest contiguous area under 16Mb, RMODE/AMODE ignored	Load begins at the largest contiguous area under 16Mb, AMODE on TEXT ESD respected unless AMODE override on LOAD command

LOAD command	System/370 mode machine	370-XA mode machine
LOAD pgma ..(RMODE ANY	Load begins at the largest contiguous area under 16Mb, RMODE/AMODE ignored	Load begins at the largest contiguous above 16Mb, AMODE on TEXT ESD respected unless AMODE override on LOAD command
LOAD pgma ..(ORIGIN TR	Load begins at start of transient area, SET LOADAREA is overridden, AMODE ignored	Load begins at start of transient area, SET LOADAREA is overridden, AMODE on TEXT ESD respected unless AMODE override on LOAD command
LOAD pgma ..(ORIGIN <i>hexloc</i>	Load begins at area specified by hexloc, SET LOADAREA is overridden, AMODE ignored	Load begins at area specified by hexloc, SET LOADAREA is overridden, AMODE on TEXT ESD respected unless AMODE override on LOAD command

Note: RMODE or AMODE ignored means that RMODE or AMODE have no effect in terms of executing the loaded program. However, if specified, they will affect the resultant RMODE or AMODE that would be passed to the GENMOD process. This allows you to develop XA programs on a System/370 mode virtual machine for execution on a 370-XA mode virtual machine.

Messages and Codes

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET NONDISP

Use the NONDISP option to define a character for CMS and the System Product Editor that is displayed in place of nondisplayable characters.

Format

SET	NONDISP [<i>char</i>]
-----	-------------------------

Operands

char

defines a character that is displayed in place of nondisplayable characters. If not specified, a blank is used.

Initial Setting

NONDISP "

Usage Notes

1. The translation of the nondisplayable character depends upon the type of terminal, whether SET APL ON or SET TEXT ON is in effect, and the current language being used (see SET LANGUAGE).
2. Changing the NONDISP setting for CMS also changes the NONDISP setting for the System Product Editor, and vice versa.
3. Changing the NONDISP character does not change characters already displayed on the screen unless that line is altered.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
 DMSSEF926E Command is only valid on display terminal [RC = 88]
 DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET NONSHARE

Use the NONSHARE option to get your own copy of a normally shared named system.

Format

SET	NONSHARE { CMSDOS CMSVSAM CMSAMS CMSBAM
------------	--

Operands

CMSDOS
CMSVSAM
CMSAMS
CMSBAM

specifies the shared named system for which you want a nonshared copy.

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]
 DMSSET400\$ System *sysname* does not exist [RC = 44]
 DMSSET401S VM size (*size*) cannot exceed *sysname* start address (*vstor*)
 [RC = 104]
 DMSSET410S Control program error indication *xxx* [RC = *rc*]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

Note: In RC = *rc*, the *rc* represents the actual error code generated by CP.

SET OUTPUT

Use the OUTPUT option to translate a hexadecimal representation displayed at the terminal to a specified character.

Format

SET	OUTPUT [xx a]
------------	----------------------

Operands

OUTPUT xx a

translates the specified hexadecimal representation xx to the specified character “a” for all xx characters displayed at the terminal.

OUTPUT

returns all characters to their default translation.

Usage Notes

1. Output translation does not occur for SCRIPT files when the SCRIPT command output is directed to the terminal, nor when you use the CMS editor on a display terminal in display mode.
2. Changing the OUTPUT setting does not translate characters already displayed on the screen unless that line is altered.
3. The OUTPUT setting does not affect trailing nulls or blanks that are at the end of a line.

Messages and Return Codes

DMSSET061E No translation character specified [RC = 24]

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET PROTECT

Use the PROTECT option to specify whether the CMS nucleus is protected against writing in its storage area.

Format

SET	PROTECT {ON OFF}
-----	---------------------

Operands

ON

protects the CMS nucleus against writing in its storage area.

OFF

does not protect the storage area containing the CMS nucleus.

Initial Setting

PROTECT ON

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET RDYMSG

Use the RDYMSG option to specify whether CMS issues the standard CMS ready message or a shortened form.

Format

SET	RDYMSG { LMSG SMSG }
-----	-------------------------

Operands

LMSG

indicates that the standard CMS ready message, including current and elapsed time, is used. The format of the standard Ready message is:

Ready; T=s.ss/s.ss hh:mm:ss

where the virtual processor time (in seconds), real processor time (in seconds), and clock time are listed.

SMSG

Indicates that a shortened form of the CMS ready message (Ready;) which does not include the time is used.

Initial Setting

RDYMSG LMSG

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET REDTYPE

Use the REDTYPE option to have CMS error messages typed in red for certain terminals equipped with the appropriate terminal feature and a two-color ribbon.

Format

SET	REDTYPE { ON } { OFF }
------------	--

Operands

ON

types CMS error messages in red for certain terminals equipped with the appropriate terminal feature and a two-color ribbon. Supported terminals are described in the *VM/SP Terminal Reference*.

OFF

suppresses red typing of error messages.

Initial Setting

REDTYPE OFF

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET RELPAGE

Use the RELPAGE option to release page frames of storage and set them to binary zeros, or to hold the pages of storage.

Format

SET	RELPAGE { ON OFF }
------------	--

Operands

ON

releases (returns to CP) unused page frames of storage and sets them to binary zeros. A page frame is considered unused whenever all of the storage allocated from it has been returned to CMS.

OFF

prevents the release of unused page frames of storage as described in SET RELPAGE ON. Use the SET RELPAGE OFF function when debugging or analyzing a problem so that the storage used is not released and can be examined. Be aware that indiscriminate use of SET RELPAGE OFF can have a negative impact on overall system performance.

Initial Setting

RELPAGE ON

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET REMOTE

Use the REMOTE option to control the display of data transmissions for CMS and the System Product Editor.

Format

SET	REMOTE { ON OFF }
------------	---

Operands

ON

specifies that data is to be compressed by removing nulls and combining data when five or more of the same characters occur consecutively in a data stream. This minimizes the amount of data transmitted and shortens the buffer, thus speeding transmission.

OFF

specifies that the data stream is not to be compressed. Data is transmitted with no minimization.

Initial Setting

REMOTE ON for remote displays

REMOTE OFF for local displays.

Usage Note

Changing the REMOTE setting for CMS also changes the REMOTE setting for the System Product Editor, and vice versa.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
 DMSSEF926E Command is only valid on a display terminal [RC = 88]
 DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET SERVER

Use the SERVER option to enable private resource processing.

Format

SET	SERVER { ON } { OFF }
------------	---

Operands

- ON**
enables incoming private resource conversation requests in the virtual machine.
- OFF**
results in the rejection of any incoming private resource conversation requests, including any which may have been queued while SET SERVER was ON.

Initial Setting

SERVER OFF

Usage Notes

1. SET SERVER ON enables APPC/VM and IUCV interrupts for private resource processing. SET SERVER OFF disables APPC/VM and IUCV interrupts unless there are any active connections or HNDIUCV SET names.
2. If the CP level is earlier than VM/SP Release 5, or if an error occurs during storage allocation, SET SERVER ON results in an error message and remains OFF.
3. If your virtual machine will be auto-logged as a result of a private resource connection request, your PROFILE EXEC must include SET SERVER ON.
4. If full-screen CMS is enabled, private resource requests are rejected even if SET SERVER is ON.
5. See the *VM/SP Connectivity Programming Guide and Reference* and the *VM/SP Connectivity Planning, Administration, and Operation* for additional information on private resource processing.

Messages and Return Codes

DMSSET1256E SET SERVER ON not allowed because CMS did not allocate a control external interrupt buffer [RC = 88]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET STORECLR

Use the SET STORECLR command to set the point of automatic GETMAIN storage cleanup and determine the action for user invocation of STRINIT.

Format

SET	STORECLR	{ ENDCMD ENDSVC }
-----	----------	----------------------

Operands

ENDCMD

indicates that CMS returns GETMAIN storage at end-of-command (the point where the CMS ready message (Ready;) is displayed) and that CMS honors user invocations of STRINIT.

ENDSVC

indicates that CMS returns GETMAIN storage when the program that invoked GETMAIN issues an SVC 202 or SVC 204 to terminate. ENDSVC also indicates that CMS treats user invocations of STRINIT as a NOP.

Initial Setting

ENDSVC

Usage Notes

1. Changing the setting causes CMS to release any GETMAIN storage currently on the SVC chain.
2. Use the QUERY STORECLR command to determine the current setting of STORECLR.
3. This command does not effect the method of GETMAIN storage cleanup at ABEND recovery. All GETMAIN storage is reclaimed at ABEND.

Messages and Return Codes

RC	Meaning
----	---------

0	Normal completion.
---	--------------------

24	An invalid parameter was specified on the SET STORECLR command.
----	---

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET SYSNAME

Use the SYSNAME option to replace a saved system name entry in the SYSNAMES table with the name of an alternate, or backup system.

Format

SET	SYSNAME { CMSDOS CMSVSAM CMSAMS CMSBAM } <i>entryname</i>
-----	--

Operands

CMSDOS *entryname*
CMSVSAM *entryname*
CMSSAMS *entryname*
CMSBAM *entryname*

allows you to replace a saved system name entry in the SYSNAMES table with the name of an alternative, or backup system. A separate SET SYSNAME command must be issued for each name entry to be changed. CMSVSAM, CMSAMS, CMSDOS, and CMSBAM are the default names assigned to the systems when the CMS system is generated.

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC=104]
DMSSET142S Saved system name *sysname* invalid [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET TEXT

Use TEXT option to activate character code conversion for TEXT characters for the System Product Editor and CMS.

Format

SET	TEXT { ON } { OFF }
-----	------------------------

Operands

ON

activates character code conversion for TEXT characters. Before using TEXT keys, issue SET TEXT ON to ensure proper character code conversion.

OFF

specifies that no character code conversion is performed for TEXT characters and keys.

Initial Setting

TEXT OFF

Usage Notes

1. The TEXT setting is valid only when performing full-screen I/O (for example, in XEDIT or in CMS with SET FULLSCREEN ON). If you are in CP or using a line-mode terminal, SET TEXT has no effect.

If you are in CP, you can issue the TERMINAL TEXT ON command to have CP convert TEXT character codes.

2. Because the text character code conversion is costly, it is recommended that you issue SET TEXT OFF when you stop using the special text keys.
3. When SET TEXT ON is specified, APL is set OFF.
4. Changing the TEXT setting for CMS also changes the TEXT setting for the System Product Editor.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
 DMSSEF524W NONDISP character reset to ".
 DMSSEF926E Command is only valid on a display terminal [RC = 88]
 DMSSET109S Virtual storage capacity exceeded [RC = 104]
 DMSWIR329W Warning: APL/TEXT option not in effect

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET THRESHOLD

SET THRESHOLD

Use the THRESHOLD option to indicate when you want a warning message or return code issued to tell you that a specified amount of your allocated file space in a file pool has been used.

Format

SET	THReshold <i>nn</i> [<i>filepoolid:</i>]
-----	--

Operands

nn

is a number, between 1 and 99 inclusive, indicating, that when this percentage of your allocated file space is used, you want a warning message.

filepoolid:

specifies in which file pool to set the threshold. If not specified, the default file pool identifier is used.

Initial Setting

THRESHOLD 90

Usage Notes

1. A default threshold percentage of 90% is set when you are enrolled with file space in a file pool.
2. Enter QUERY LIMITS * to obtain information about your usage of the file space available to you, and the threshold setting.
3. The warning message or return code will continue to be issued whenever you enter a command that affects your file space until you are below the threshold. For example, you receive the threshold warning and then erase a file in your file space. If you are still at or above the threshold level, you will receive another threshold warning.
4. If the SET THRESHOLD command is issued from an exec on a work unit that is active, the command will fail.
5. If you are using OS simulation, the user threshold warning is treated differently than the rest of CMS. The user threshold warning for OS is treated as a disk full error. This could take place on a write or put operation.

Example

If you have been allocated 1000 blocks of file space and you enter:

```
set threshold 95
```

the threshold is set at 95%. You will get a warning message when 950 or more blocks of your default file pool file space are used.

Messages and Return Codes

DMSJSE1140E You are not enrolled in file pool *filepoolid* [RC=40]
DMSJSE1141W User filespace threshold still exceeded for the file pool *filepoolidid*
[RC=4]
DMSJSE1168E Invalid threshold value *value* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

SET TRANSLATE

Use the SET TRANSLATE command to suppress translations and translation synonyms of command names for a language. The SET TRANSLATE command specifies the way in which the User Language Translation Tables and the System National Language Translation Tables are used.

Format

SET	TRANslate { ON } { OFF }	[SYStem USER ALL]	[TRANslate SYNonym BOTH]	[APPLID <i>applid</i>] * -]
-----	-----------------------------	---------------------------------	--	--

Operands

- ON**
allows you to specify the table to be used for command name translation.
- OFF**
allows you to specify that a table will not be used for command name translation.
- SYStem**
translates command names using only the System National Language Translation Table.
- USER**
translates command names using only the User National Language Translation Table.
- ALL**
translates command names using both the User and System National Language Translation Tables.
- TRANslate**
indicates that only the national language translations are set ON or OFF.
- SYNonym**
indicates that only the national language translation synonyms are set ON or OFF.
- BOTH**
indicates that both national language translations and translation synonyms are set ON or OFF.
- APPLid *applid***
specifies the application for which a table is to be set ON or OFF. It must be three alphanumeric characters, and the first character must be alphabetic. The default, *, sets the tables for all applications.

Usage Notes

1. If you issue the SET command specifying an invalid function and the implied CP function is in effect, you may receive message DMKCF003E INVALID OPTIONS - option.
2. If an invalid SET command function is specified from an exec and the implied CP function is in effect, then the return code is -0003.
3. To determine or verify the setting of most functions, use the QUERY command.
4. Translation synonyms cannot be set ON unless translations are also set ON. Likewise, translations cannot be set OFF unless translation synonyms are also set OFF.
5. The settings for the TRANSLATE operand are enabled in the following order: System Translations, System Translation Synonyms, User Translations, and User Translation Synonyms.

Example

To set the translation synonym table OFF for all applications, enter
 set translate off user syn

Messages and Return Codes

DMSSET109S Virtual storage capacity exceeded [RC = 104]
 DMSSET258E {user|system} translation synonyms can not be set ON unless
 {user|system} translations are also set ON, application id: *applid*
 [RC = 28]
 DMSSET258E {user|system} translations can not be set OFF unless {user|system}
 translation synonyms are also set OFF, application id: *applid*
 [RC = 28]
 DMSSET280E Application *applid* not active [RC = 28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET UPSI

Use the UPSI option in the CMS/DOS environment to set the User Program Switch Indicator (UPSI) byte to the specified bit string of 0's and 1's, or to reset the UPSI byte to binary zeros.

Format

SET	UPSI { <i>nnnnnnnn</i> OFF }
------------	--

Operands

nnnnnnnn

sets the UPSI (User Program Switch Indicator) byte to the specified bit string of 0's and 1's. If you enter fewer than eight digits, the UPSI byte is filled in from the left and zero-padded to the right. If you enter an "x" for any digit, the corresponding bit in the UPSI byte is left unchanged.

OFF

resets the UPSI byte to binary zeros.

Initial Setting

UPSI OFF

Messages and Return Codes

DMSSET099E CMS/DOS environment not active [RC = 40]

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SETKEY

Use the SETKEY command to change CMS storage keys.

Format

SETKEY	<i>key sysname [startadr]</i>
--------	-------------------------------

Operands

key

is the storage protection key, specified in decimal. Valid keys are 0-15.

sysname

is the name of the saved system or segment for which the storage protection is being assigned.

startadr

is the starting address (in hexadecimal) at which the keys are to be assigned. The address must be within the address range defined for the saved system or discontinuous saved segments. Using the **startadr** operand, you can issue the SETKEY command several times and, thus, assign different keys to various portions of the saved system or segment.

SETPRT

Use the SETPRT command to load virtual 3800 printers. The SETPRT command is valid only for a virtual 3800 Model 1 or 3800 Model 3 printer.

Format

SETPRT	<pre> Chars [(]cccc...[)] COpies [(]nnn[)] COPYnr [(]nnn[)] Fcb [(]ffff [)] FLash [(]id nnn [)] Init Modify [(]mmmm [n][)] </pre>
---------------	--

Operands

Chars *cccc...*

specifies the names of from one to four character arrangement tables (CATs) to be loaded into the virtual 3800 printer. CAT names may be from one to four alphameric characters. The CATs must exist as 'XTB1cccc TEXT' files on an accessed CMS disk or SFS directory.

COpies *nnn*

specifies the total number of copies of each page to be printed. The value of nnn must be a number from 1 to 255. The default value is 1.

COPYnr *nnn*

specifies the copy number of the first copy in a copy group. The value of nnn must be a number from 1 to 255. If COPYNR is not specified, a starting copy number of 1 is assumed.

Fcb *ffff*

specifies the FCB to be loaded into the virtual 3800 printer. The FCB must exist as a 'FCB3ffff TEXT' file on an accessed CMS disk or SFS directory, unless ffff is specified as 6, 8, 12, or for the 3800 Model 3 printer only, 10. In that case, the FCB is not loaded from a CMS file. CP determines the appropriate FCB to load and prints the entire file at 6, 8, 12, or for the 3800 Model 3 printer, 10 lines per inch.

FLash *id nnn*

specifies the one- to four-character overlay name (id) and the number of copies of each page (nnn) to be printed with the overlay indicated by 'id'. The value of nnn may be a number from 0 to 255. If nnn is not specified, 1 is the default. If the FLASH keyword is omitted, no copies are printed with an overlay.

Init

specifies that an "Initialize Printer" CCW will be issued before any other functions specified in this command are performed.

Modify *mmmm* [*n*]

specifies copy modification data to be loaded. The copy modification must exist as a 'MOD1mmmm TEXT' file on an accessed CMS disk or SFS directory. Further, *n* specifies the CAT to use for the copy modification load. If *n* is omitted, 0 is the default.

Note: Keyword values must be enclosed in parentheses only if they could be interpreted as a SETPRT keyword or keyword abbreviation. Otherwise the parentheses may be omitted.

Usage Notes

1. In the Shared File System, an alias can be used to refer to a TEXT file used by SETPRT only if the alias was used when the TEXT file was created.

The name of the alias must conform to the naming conventions of the TEXT file. For example, an FCB must exist as FCB3ffff TEXT.
2. CATs must be specified so that they correspond to the appropriate TRC bytes. The first CAT specified corresponds to TRC byte 0, the second CAT corresponds to TRC byte 1, and so on.
3. CATs can reference the Library Character Set (LCS) modules and Graphic Character Modification Modules (GRAPHMODS) for both the 3800 Model 1 and Model 3 printers. SETPRT uses naming conventions to select the correct modules for the defined 3800 model printer.
4. Customized 3800 Model 1 character sets must be converted from the 180 x 144 to the 240 x 240 pel density format before they may be used in a 3800 Model 3 printer.
5. If the number of copies specified with the FLASH keyword is greater than the number of copies specified in COPIES nnn, the actual number of copies printed will equal the number specified with the FLASH keyword. Thus, if you want all copies to be printed with an overlay, you can specify the number with the FLASH keyword and omit the COPIES keyword.
6. The use of 'INIT' and 'FCB 6|8|10|12' together causes the printer to always be reset to 6 lines per inch. Both the INIT CCW and the 'CP SPOOL 00E FCB 6|8|10|12' generated by the 'FCB 6|8|10|12' are passed to CP. The LOADFCB CCW is sent to the printer before the INIT CCW. This resets the FCB to the Init IMPL Default of 6 lines per inch. 'INIT' and 'FCB ffff' do not have this problem, since 'FCB ffff' is handled directly by CMS.
7. SETPRT FCB 6|8|10|12 sets the FCB of the virtual printer until it is specifically changed. SETPRT FLASH sets the FLASH until another SETPRT is issued. All other SETPRT options only affect the next file to be printed.
8. The length of the FCB to be loaded must agree with the forms length specified in the SIZE parameter of the CP DEFINE 3800 command.

Example

For example, to indicate that you want to use character set GF10 printed at 6 lines per inch, enter:

```
setprt chars gf10 fcb 6
```

SETPRT

Responses

DMSSPR196I Printer *vdev* setup complete

The virtual 3800 printer was successfully loaded.

Messages and Return Codes

DMSERD257T Internal system error at address *address* (offset *offset*)
DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
DMSSPR002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
DMSSPR014E Invalid keyword *function* [RC=24]
DMSSPR026E Invalid value *value* for keyword *keyword* [RC=24]
DMSSPR113S Printer 00E not attached [RC=100]
DMSSPR145S Intervention required on printer [RC=100]
DMSSPR197S Undiagnosed error from printer 00E [RC=100]
DMSSPR198E SETPRT load check; sense = *sense*
DMSSPR199S Printer 00E not a virtual 3800 Model 1 or Model 3
DMSSPR204E Too many WCGMs needed for CHARS
DMSSPR352E Invalid SETPRT data in file *fn ft*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

SORT

Use the SORT command to read fixed-length records from a CMS input file, arrange them in ascending EBCDIC order according to specified sort fields, and create a new file containing the sorted records.

Format

SORT	<i>fileid1 fileid2</i>
-------------	------------------------

Operands*fileid1*

is the file identifier (file name, file type, file mode) of the file containing the records to be sorted.

fileid2

is the file identifier (file name, file type, file mode) of the new output file to contain the sorted records.

Usage Note

The input and output files must not have the same file identifiers, since SORT cannot write the sorted output back into the space occupied by the input file. If *fileid2* is the same as *fileid1*, message

DMSRT019E Identical fileids

is issued and the SORT operation does not take place. If *fileid1* and *fileid2* are different and a file with the same name as *fileid2* already exists, the existing file is replaced when the SORT operation takes place.

Entering Sort Control Fields: After the SORT command is entered, CMS responds with the following message on the terminal:

DMSRT604R Enter sort fields:

Respond by entering one or more pairs of numbers of the form “xx yy”; separate each pair by one or more blanks. Each “xx” is the starting character position of a sort field within each input record and “yy” is the ending character position. The leftmost pair of numbers denotes the major sort field. The number of sort fields is limited to the number of fields you can enter on one line. The records can be sorted on up to a total of 253 positions.

Virtual Storage Requirements for Sorting: The sorting operation takes place with two passes of the input file. The first pass creates an ordered pointer table in virtual storage. The second pass uses the pointer table to read the input file in a random manner and write the output file. Therefore, the size of storage and the size and number of sort fields are the limiting factors in determining the number of records that can be sorted at any one time.

SORT

Responses

DMSRT604R Enter sort fields:

You are requested to enter SORT control fields. You should enter them in the form described previously in "Entering Sort Control Fields."

Messages and Return Codes

DMSOPN002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
DMSRT009E Column *col* exceeds record length [RC=24]
DMSRT019E Identical fileids [RC=24]
DMSRT034E File *fn ft fm* is not fixed length [RC=32]
DMSRT037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
DMSRT053E Invalid sort field pair defined [RC=24]
DMSRT054E Incomplete fileid specified [RC=24]
DMSRT062E Invalid * in fileid [*fn ft fm*] [RC=20]
DMSRT063E No list entered [RC=40]
DMSRT069E Filemode *mode* not accessed [RC=36]
DMSRT070E Invalid parameter *parameter* [RC=24]
DMSRT104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
DMSRT105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
DMSRT212E Maximum number of records exceeded [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

SSERV

Use the SSERV command in CMS/DOS to copy, display, print, or punch a book from a VSE source statement library.

Format

SSERV	<i>sublib bookname</i> [<i>ft</i> COPY] [(options...)]] <u>Options:</u> [DISK] [PRINT] [PUNCH] [TERM]
--------------	---

Operands

sublib

specifies the source statement sublibrary in which the book is cataloged.

bookname

specifies the name of the book in the VSE private or system source statement sublibrary. The private library, if any, is searched before the system library.

ft

specifies the file type of the file to be created on your disk or directory accessed as A. "ft" defaults to COPY if a file type is not specified. The file name is always the same as the book name.

Options

You may enter as many options as you wish, depending upon the functions you want to perform.

DISK

copies the book to a CMS file.

PUNCH

punches the book on the virtual punch.

PRINT

spools a copy of the book to your virtual printer.

TERM

displays the book on your terminal.

Usage Notes

1. If you want to copy books from private libraries, you must issue an ASSGN command for the logical unit SYSSLB and identify the library on a DLBL command line using a ddname of IJSYSSL.

If you want to copy books from the system library, you must have entered the CMS/DOS environment specifying the mode letter of the system residence volume.

SSERV

2. You should not use the SSERV command to copy books from macro (E) sublibraries, since they are in "edited" (that is, compressed) form. Use the ESERV command to copy and de-edit macros from a macro (E) sublibrary.

Responses

When you use the TERM option, the specified book is displayed at the terminal.

Messages and Return Codes

DMSSRV003E	Invalid option: <i>option</i> [RC=24]
DMSSRV004E	Book <i>subl.book</i> not found [RC=28]
DMSSRV006E	No read/write A filemode accessed [RC=36]
DMSSRV070E	Invalid parameter <i>parameter</i> [RC=24]
DMSSRV089E	Open error code <i>nn</i> on SYSSLB [RC=36]
DMSSRV097E	No SYSRES volume active [RC=36]
DMSSRV098E	No book name specified [RC=24]
DMSSRV099E	CMS/DOS environment not active [RC=40]
DMSSRV105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC=100]
DMSSRV113S	Disk(<i>vdev</i>) not attached [RC=100]
DMSSRV194S	Book <i>subl.book</i> contains bad records [RC=100]
DMSSRV411S	Input error code <i>nn</i> on SYSaaa [RC= <i>rc</i>]

START

Use the START command to begin execution of CMS, OS, or VSE programs that were previously loaded or fetched.

Format

START	$\left[\begin{array}{l} \textit{entry} \text{ [args...]} \\ * \\ \text{(option [])} \end{array} \right]$ <p><u>Option:</u> [NO]</p>
--------------	--

Operands

entry

passes control to the control section name or entry point name at execution time. The operand, *entry*, may be a file name only if the file name is identical to a control section name or an entry point name.

*

passes control to the default entry point. See the discussion of the LOAD command for a discussion of the default entry point selection.

args...

are arguments to be passed to the started program. If user arguments are specified, the *entry* or * operands must be specified; otherwise, the first argument is taken as the entry point. Arguments are passed to the program via general register 1. The *entry* operand and any arguments become a string of doublewords, one argument per doubleword, and the address of the list is placed in general register 1.

Option

NO

suppresses execution of the program. Linkage editor and loader functions are performed and the program is in storage ready to execute, but control is not given to the program. START * and START (NO) are mutually exclusive.

Usage Notes

1. Any undefined names or references specified in the files loaded into storage are defined as zero. Thus, if there is a call or branch to a subroutine from a main program, and if the subroutine has never been loaded, the call or branch transfers control at execution time to location zero of the virtual machine.
2. Do not use the START command for programs that are generated via the GENMOD command with the NOMAP option. The START command does not execute properly for such programs.

START

- When arguments are passed on the START command, the requirements of both CMS and the language of the application program must be met. For example, COBOL programs require arguments separated by commas:

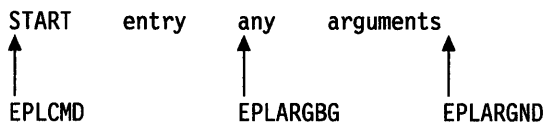
```
START * A,B,C
```

See the appropriate language guide for details on parameter requirements.

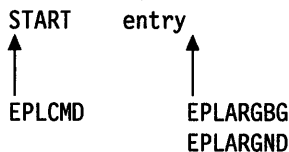
- Issue the START command immediately following the LOAD and INCLUDE commands. If the LOAD and INCLUDE were issued in an exec procedure, issue the START command from within the exec as well.
- If START is issued from the virtual console or from an EXEC 2 EXEC, register 0 points to an extended parameter list block. The extended parameter list for the START command pointed to by register 0 has the following structure:

```
DC    A(EPLCMD)
DC    A(EPLARGBG)
DC    A(EPLARGGND)
DC    A(0)
```

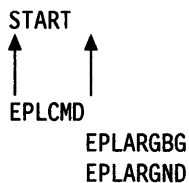
where:



or:



or:



Responses

DMSLI0740I Execution begins ...

is displayed when the designated entry point is validated.

This message is suppressed if CMS/DOS is active and the COMP option is specified in the FETCH command.

Messages and Return Codes

DMSLIO021E	Entry point <i>name</i> not found [RC=40]
DMSLIO055E	No entry point defined [RC=40]

STATE/STATEW (ESTATE/ESTATEW)

Use the STATE or ESTATE command to verify the existence of a CMS, OS, or DOS file that may reside on any accessed disk or accessed Shared File System (SFS) directory. Use the STATEW or ESTATEW command to verify the existence of a file residing on a read/write disk or read/write SFS directory.

It may be necessary to use ESTATE/ESTATEW when writing execs or assembler programs. This enables STATE to handle files larger than 65533 records. (This is the equivalent of coding the FSSTATE macro with the FORM = E option.)

Format

STATE STATEW ESTATE ESTATEW	$\left\{ \begin{array}{c} fn \\ * \end{array} \right\} \left\{ \begin{array}{c} ft \\ * \end{array} \right\} \left[\begin{array}{c} fm \\ * \\ - \end{array} \right]$
--------------------------------------	--

Operands

fn

is the file name of the file whose existence is to be verified. If an asterisk is specified, the first file found satisfying the rest of the file ID is used.

ft

is the file type of the file whose existence is to be verified. If an asterisk is specified, the first file found satisfying the rest of the file ID is used.

fm

is the file mode of the file whose existence is to be verified. If *fm* is specified, the parent disk or directory and its read-only extensions will be searched. If *fm* is omitted, or specified with an asterisk, all your accessed disks or directories (A-Z) are searched.

Usage Notes

1. If you issue the STATEW command specifying a file that exists on a read-only disk or directory, you receive error message DMSSTT002E or file not found.
2. For data stored in SFS directories, STATE will not find subdirectories, erased or revoked aliases, or files for which you do not have read or write authority.
3. When you code an asterisk for *fn* or *ft*, the search for the file is ended as soon as any file satisfies any of the other conditions. For example,

```
state * file
```

executes successfully if any file on any accessed disk or directory (including the system disk) has a file type of FILE.

4. To verify the existence of an OS or VSE file when DOS is set OFF, you must issue the FILEDEF command to establish a CMS file identifier for the file. For example, to verify the existence of the OS file TEST.DATA on an OS disk accessed as C you could enter:

```
filedef check disk check list c dsn test data
state check list
```

where CHECK LIST is the CMS file name and file type associated with the OS data set name.

- To verify the existence of an OS or VSE file when the CMS/DOS environment is active, you must issue the DLBL command to establish a CMS file identifier for the file. For example, to verify the existence of the DOS file TEST.DATA on a DOS disk at file mode C, you could enter:

```
dlbl check c dsn test data
state file check
```

where FILE CHECK is the default CMS file name and file type (FILE ddname) associated with the VSE file ID.

- The STATE command disregards the file mode number specified when both the file name and file type are explicitly specified. When the file name or file type (or both) are specified as asterisk (*), the file mode number is used.
- To verify the syntax of a file identifier (file name, file type, file mode) without verifying the existence of the file, use the VALIDATE command.
- You can invoke the STATE/STATEW command from the terminal, from an exec file, or as a function from a program. If STATE/STATEW is invoked as a function or from an exec file that has the message output suppressed, the message DMSSTT002E File *fn ft fm* not found is not issued.
- If the STATE/STATEW (or ESTATE/ESTATEW) command is invoked via SVC204 in an assembler program, the address of the STATEFST copy is returned at X'1C' into the STATE parameter list.

Responses

The CMS ready message indicates that the specified file exists.

```
DMSSTT227I Processing volume nn in data set data set name
```

The specified data set has multiple volumes; the volume being processed is shown in the message. The STATE command treats end-of-volume as end-of-file and there is no end-of-volume switching.

```
DMSSTT228I User labels bypassed on data set data set name
```

The specified data set has disk user labels; these labels are skipped.

Messages and Return Codes

```
DMSSTT002E File [fn [ft [fm]]] not found [RC = 28]
DMSSTT048E Invalid mode mode [RC = 24]
DMSSTT054E Incomplete fileid specified [RC = 24]
DMSSTT062E Invalid character char in fileid fn ft [RC = 20]
DMSSTT062E SO and SI are invalid fileid characters [RC = 20]
DMSSTT069E Filemode mode not accessed [RC = 36]
DMSSTT070E Invalid parameter parameter [RC = 24]
DMSSTT229E Unsupported OS data set, error nn [RC = 8n]
DMSSTT253E File fn ft fm cannot be handled with supplied parameter list
[RC = 88]
```

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SVCTRACE

Use the SVCTRACE command to trace and record information about supervisor calls occurring in your virtual machine.

Format

SVCTrace	<div style="display: inline-block; vertical-align: middle;"> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">ON</div> <div style="border: 1px solid black; padding: 2px;">OFF</div> </div>
-----------------	--

Operands**ON**

starts tracing all SVC instructions issued within CMS.

OFF

stops SVC tracing.

Usage Notes

1. The trace information recorded on the printer includes:
 - The virtual storage location of the calling SVC instruction and the name of the called program or routine
 - The normal and error return addresses
 - The contents of the general registers both before the SVC-called program is given control and after a return from that program
 - The contents of the general registers when the SVC handling routine is finished processing
 - The contents of the floating-point registers before the SVC-called program is given control and after a return from that program
 - The contents of the floating-point registers when the SVC handling routine is finished processing
 - The parameter list passed to the SVC
2. To terminate tracing previously established by the SVCTRACE command, issue the HO or SVCTRACE OFF commands. SVCTRACE OFF and HO cause all trace information recorded, up to the point they are issued, to be printed on the virtual spooled printer. On typewriter terminals SVCTRACE OFF can be issued only when the keyboard is unlocked to accept input to the CMS command environment. To terminate tracing at any other point in system processing, HO must be issued. To suspend tracing temporarily during a session, interrupt processing and enter the Immediate command SO (Suspend Tracing). To resume tracing that was suspended with the SO command, enter the Immediate command RO (Resume Tracing).

If you issue the CMS Immediate command HX or you log off the system before termination of tracing previously set by the SVCTRACE command, the switches are cleared automatically and all recorded trace information is printed on the virtual spooled printer.

3. If a user timer exit is activated while SVCTRACE is active, SVCTRACE is disabled for the duration of the timer exit. Any SVCs issued during the timer exit are not reflected in the SVCTRACE listing.
4. If your program must remain disabled for interrupts (in an interrupt handler, for example), it must not issue any SVC's while SVCTRACE is active. SVCTRACE enables the system for interrupts. Use the CP PER command instead.
5. When tracing on a virtual machine with only one printer, the trace data is intermixed with other data sent to the virtual printer.

Responses

A variety of information is printed whenever the:

svctrace on

command is issued.

The first line of trace output starts with a dash or plus sign or an asterisk (- or + or *). The format of the first line of trace output is:

```
- N/D = xxx/dd name FROM loc OLDPSW = psw1 GOPSW = psw2 [RC=rc]
+
*
```

where:

- indicates information recorded before processing the SVC.
- + indicates information recorded after processing the SVC, unless the asterisk (*) applies.
- * indicates information recorded after processing a CMS SVC that had an error return.

N/D is an abbreviation for SVC number and depth (or level).

xxx is the number of the SVC call (they are numbered sequentially).

dd is the nesting level of the SVC call.

name is the macro or routine being called.

loc is the program location from which the SVC was issued.

psw1 is the PSW at the time the SVC was called.

psw2 is the PSW with which the routine being called is invoked, if the first character of this line is a dash (-). If the first character of this line is a plus sign or asterisk (+ or *), PSW2 represents the PSW that returns control to the user.

rc is the return code from the SVC handling routine in general register 15. This field is omitted if the first character of this line is a dash (-), or if this is an OS SVC call. For a CMS SVC, this field is 0 if the line begins with a plus sign (+), and nonzero for an asterisk (*). Also, this field equals the contents of R15 in the "GPRS AFTER" line.

The next two lines of output are the contents of the general registers when control is passed to the SVC handling routine. This output is identified at the left by ".GPRSB." The format of the output is:

```
.GPRSB = h h h h h h h h *ddddddd*
        = h h h h h h h h *ddddddd*
```

where *h* represents the contents of a general register in hexadecimal format and *d* represents the EBCDIC translation of the contents of a general register. The contents of general registers 0 through 7 are printed on the first line, with the contents of registers 8 through F on the second line. The hexadecimal contents of the registers are printed first, followed by the EBCDIC translation. The EBCDIC translation is preceded and followed by an asterisk(*).

The next line of output is the contents of general registers 0, 1, and 15 when control is returned to your program. The output is identified at the left by “.GPRS AFTER :.” The format of the output is:

```
.GPRS AFTER : R0-R1 = h h *dd* R15 = h *d*
```

where *h* represents the hexadecimal contents of a general register and *d* is the EBCDIC translation of the contents of a general register. The only general registers that CMS routines alter are registers 0, 1, and 15 so only those registers are printed when control returns to your program. The EBCDIC translation is preceded and followed by an asterisk (*).

The next two lines of output are the contents of the general registers when the SVC handling routine is finished processing. This output is identified at the left by “.GPRSS.” The format of the output is:

```
.GPRSS = h h h h h h h h *ddddddd*
        = h h h h h h h h *ddddddd*
```

where *h* represents the hexadecimal contents of a general register and *d* represents the EBCDIC translation of the contents of a general register. General registers 0 through 7 are printed on the first line with registers 8 through F on the second line. The EBCDIC translation is preceded and followed by an asterisk (*).

The next line of output is the contents of the calling routine's floating-point registers. The output is identified at the left by “.FPRS.” The format of the output is:

```
.FPRS = f f f f *gggg*
```

where *f* represents the hexadecimal contents of a floating-point register and *g* is the EBCDIC translation of a floating-point register. Each floating point register is a doubleword; each *f* and *g* represents a doubleword of data. The EBCDIC translation is preceded and followed by an asterisk (*).

The next line of output is the contents of floating-point registers when the SVC handling routine is finished processing. The output is identified by “.FPRSS” at the left. The format of the output is:

```
.FPRSS = f f f f *gggg*
```

where *f* represents the hexadecimal contents of a floating-point register and *g* is the EBCDIC translation. Each floating-point register is a doubleword and each *f* and *g* represents a doubleword of data. The EBCDIC translation is preceded and followed by an asterisk (*).

The last two lines of output are printed only if the address in register 1 is a valid address for the virtual machine. If printed, the output is the parameter list passed to the SVC. The output is identified by “.PARM” at the left. The output format is:

```
.PARM = h h h h h h h h *ddddddd*
        = h h h h h h h h *ddddddd*
```

where *h* represents a word of hexadecimal data and *d* is the EBCDIC translation. The parameter list is found at the address contained in register 1 before control is

passed to the SVC handling program. The EBCDIC translation is preceded and followed by an asterisk (*).

Table 16 summarizes the types of SVC trace output.

Table 16. Summary of SVCTRACE Output Lines	
Identification	Comments
- N/D	The SVC and the routine that issued the SVC.
(+) N/D	The SVC and the routine that issued the SVC.
(*) N/D	The SVC and the routine that issued the SVC.
.GPRSB	Contents of general registers when control is passed to the SVC handling routine.
.GPRS AFTER	Contents of general registers 0, 1, and 15 when control is returned to your program.
.GPRSS	Contents of the general registers when the SVC handling routine is finished processing.
.FPRS	Contents of floating-point registers before the SVC-called program is given control and after returning from that program.
.FPRSS	Contents of the floating-point registers when the SVC handling routine is finished processing.
.PARM	The parameter list, when one is passed to the SVC.

Messages and Return Codes

- DMSERD257T Internal system error at address *addr* (offset *offset*)
- DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC = 31]
- DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
- DMSOVR014E Invalid function *function* [RC = 24]
- DMSOVR047E No function specified [RC = 24]
- DMSOVR104S Error *nn* reading file *fn ft fm* from disk or directory [RC = 31|55|70|76|99|100]
- DMSOVR109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

SYNONYM

Use the SYNONYM command to invoke a table of synonyms to be used with, or in place of, CMS and user-written command names. You create the table yourself using the editor. To specify entries for the table, see the section after the Usage Notes called The User Synonym Table.

The names you define can be used either instead of or in conjunction with the standard CMS command truncations. However, no matter what truncations, synonyms, or truncations of the synonyms are in effect, the full real name of the command is always accepted.

Format

SYNONym	$\left[\begin{array}{c} fn \\ \left[\begin{array}{c} \text{SYNONYM} \\ \left[\begin{array}{c} fm \\ A1 \\ * \end{array} \right] \end{array} \right] \end{array} \right] \left[(\text{options...}) \right]$ <p style="text-align: center;"><u>Options:</u> $\left[\begin{array}{c} \text{STD} \\ \text{NOSTD} \end{array} \right] \quad [\text{CLEAR}]$</p>
----------------	---

Operands

fn

is the file name of the file containing your synonyms table.

fm

is the file mode of the file containing your synonyms; if omitted, your disk or directory accessed as A and its extensions are searched. If you specify *fm*, you must enter the keyword, SYNONYM. If you specify *fm* as an asterisk (*), all accessed disks and directories are searched for the specified SYNONYM file.

Options

STD

specifies that standard CMS abbreviations are accepted.

NOSTD

standard CMS abbreviations are not to be accepted. (The full CMS command and the synonyms you defined can still be used.)

CLEAR

removes any synonym table set by a previously entered SYNONYM command.

Usage Notes

1. If you enter the SYNONYM command with no operands, the system synonym table and the user synonym table (if one exists) are listed.
2. The SET ABBREV ON or OFF command, in conjunction with the SYNONYM command, determines which standard and user-defined forms of a particular CMS command are acceptable.

3. The SYNONYM command cannot define national language translations.
4. Exec procedures having a synonym defined for them can be invoked by its synonym if implied exec (IMPEX) function is on. However, within an exec procedure, only the exec file name can be used. A synonym is not recognized within an exec since the synonym tables are not searched during exec processing.
5. When ABBREV is OFF, the synonyms defined for command names (EXECs and MODULEs) are valid, but abbreviations are ignored.

The User Synonym Table

You create the synonym table using the CMS editor. The table must be a file with the file type SYNONYM. The file consists of 80-byte fixed-length records in free-form format with columns 73-80 ignored. The format for each record is:

```
systemcommand usersynonym count
```

where:

systemcommand

is the name of the CMS command or MODULE or exec file for which you are creating a synonym.

usersynonym

is the synonym you are assigning to the command name. When you create the synonym, you must follow the same syntax rules as for commands; that is, you must use the character set used to create commands, the synonym may be no longer than eight characters, and so on.

count

is the minimum number of characters that must be entered for the synonym to be accepted by CMS. If omitted, the entire synonym must be entered (see the following example).

Note: By default CMS translates commands entered on the command line into uppercase. Therefore, you should enter your synonyms in the table in uppercase.

A table of command synonyms is built from the contents of this file. You may have several synonym files but only one may be active at a time. For example, if the synonym file named MYSYN contains:

```
MOVEFILE MVIT
```

then, after you have issued the command:

```
synonym mysyn
```

the synonym MVIT can be entered as a command name to execute the MOVEFILE command. It cannot be truncated since no count is specified. If MYSYN SYNONYM contains:

```
ACCESS GETDISK 3
```

then, the synonyms GET, GETD, GETDI, GETDIS, or GETDISK can be entered as the command name instead of ACCESS.

If you have an exec file named TDISK, you might have a synonym entry:

```
TDISK TDISK 2
```

so that you can invoke the exec procedure by specifying the truncation TD.

SYNONYM

The Relationship between the SET ABBREV and SYNONYM Commands: The default values of the SET and SYNONYM commands are such that the system synonym abbreviation table is available unless otherwise specified.

The system synonym abbreviation table for the FILEDEF command states that FI is the minimum truncation. Therefore, the acceptable abbreviations for FILEDEF are: FI, FIL, FILE, FILED, FILEDE, and FILEDEF. The system synonym abbreviation table is available whenever both SET ABBREV ON and SYNONYM (STD) are in effect.

If you have a synonym table with the file identification USERTAB SYNONYM A, that has the entry:

FILEDEF USERNAME 3

then, USERNAME is a synonym for FILEDEF, and acceptable truncations of USERNAME are: USE, USEN, USENA, USENAM, and USERNAME. The user synonym abbreviation table is available whenever both SET ABBREV ON and SYNONYM USERTAB are specified.

No matter what synonyms and truncations are defined, the full real name of the command is always in effect.

Table 17 lists the forms of the system command and user synonyms available for the various combinations of the SET ABBREV and SYNONYM commands.

Options	Acceptable Command Forms	Comments
SET ABBREV ON SYN USERTAB (STD)	FI FIL : FILEDEF USE USEN : USERNAME	The ABBREV ON option of the SET command and the STD option of the SYNONYM command make the system table available. The user synonym, USERNAME, is available because the synonym table (USERTAB) is specified on the SYNONYM command. The truncations for USERNAME are available because SET ABBREV ON was specified with the USERTAB also available.
SET ABBREV OFF SYN USERTAB (STD)	FILEDEF USERNAME	The user-defined synonym, USERNAME, is permitted because the user synonym table (USERTAB) is specified on the SYNONYM command. No system or user truncations are permitted.
SET ABBREV ON SYN USERTAB (NOSTD)	FILEDEF USE USEN : USERNAME	The system synonym table is unavailable because the NOSTD option is specified on the SYNONYM command. The user synonym, USERNAME, is available because the user synonym table (USERTAB) is specified on the SYNONYM command and the truncations of USERNAME are permitted because SET ABBREV ON is specified with USERTAB also available.
SET ABBREV OFF SYN USERTAB (NOSTD)	FILEDEF USERNAME	The system synonym table is made unavailable either by the SET ABBREV OFF command or by the SYN (NOSTD) command. The synonym, USERNAME, is permitted because the user-defined synonym table (USERTAB) is specified on the SYNONYM command. The truncations for USERNAME are not permitted because the SET ABBREV OFF option is in effect.
SET ABBREV ON SYN (CLEAR STD)	FI FIL : FILEDEF	The user-defined table is now unavailable. The system synonym table is available because both the ABBREV ON option of the SET command and the STD option of the SYNONYM command are specified.

Table 17 (Page 2 of 2). System and User-Defined Truncations

Options	Acceptable Command Forms	Comments
SET ABBREV OFF SYN (CLEAR STD SET ABBREV ON SYN (CLEAR NOSTD SET ABBREV OFF SYN (CLEAR NOSTD	FILEDEF	Because CLEAR is specified on the SYNONYM command, the synonym and its truncations are no longer available. Either the SET ABBREV OFF command or the SYNONYM (NOSTD command make the system synonym table unavailable.

This table does not apply to execs.

Responses

When you enter the SYNONYM command with no operands, the synonym table(s) currently in effect are displayed.

SYSTEM COMMAND	USER SYNONYM	SHORTEST FORM (IF ANY)
.	.	.
.	.	.
.	.	.

This response is the same as the response to the command QUERY SYNONYM ALL.

DMSSYN711I No system synonyms in effect

This response is displayed when you issue the SYNONYM command with no operands after the command SYNONYM (NOSTD) has been issued.

DMSSYN712I No synonyms (DMSINA not in nucleus)

The system routine which handles SYNONYM command processing is not in the system.

Messages and Return Codes

DMSERD257T	Internal system error at address <i>addr</i> (offset <i>offset</i>)
DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC=31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSSYN002E	File [<i>fn</i> [<i>ft</i> [<i>fm</i>]]] not found [RC=28]
DMSSYN003E	Invalid option: <i>option</i> [RC=24]
DMSSYN007E	File <i>fn ft fm</i> [is] not fixed, 80-character records [RC=32]
DMSSYN032E	Invalid filetype <i>ft</i> [RC=24]
DMSSYN056E	File <i>fn ft [fm]</i> contains invalid record formats [RC=32]
DMSSYN066E	<i>option1</i> and <i>option2</i> are conflicting options [RC=24]
DMSSYN104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC=31 55 70 76 99 100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814

TAPE

Use the TAPE command to dump CMS-formatted files from a disk or directory to tape, load previously dumped files from tape to a disk or directory, and perform various control operations on a specified tape drive. Files processed by the TAPE command must be in a unique CMS format. The TAPE command does not process multi-volume files. Files to be dumped can contain either fixed- or variable-length records.

Format

TAPE	<div style="border: 1px solid black; padding: 10px; margin-bottom: 20px;"> <p>DUMP $\left\{ \begin{matrix} fn \\ * \end{matrix} \right\} \left\{ \begin{matrix} ft \\ * \end{matrix} \right\} \left[\begin{matrix} fm \\ * \end{matrix} \right]$ [(optionA optionB optionD optionF [])]</p> <p>LOAD $\left[\left\{ \begin{matrix} fn \\ * \end{matrix} \right\} \left\{ \begin{matrix} ft \\ * \end{matrix} \right\} \left[\begin{matrix} fm \\ A \end{matrix} \right] \right]$ [(optionB optionC optionD [])]</p> <p>SCAN $\left[\left\{ \begin{matrix} fn \\ * \end{matrix} \right\} \left\{ \begin{matrix} ft \\ * \end{matrix} \right\} \right]$ [(optionB optionC optionD [])]</p> <p>SKIP $\left[\left\{ \begin{matrix} fn \\ * \end{matrix} \right\} \left\{ \begin{matrix} ft \\ * \end{matrix} \right\} \right]$ [(optionB optionC optionD [])]</p> <p>DVOL1 [(optionD optionE [])]</p> <p>WVOL1 <i>valid</i> {owner} [(optionD optionE [])]</p> <p>MODESET [(optionD [])]</p> <p><i>tapcmd</i> $\left[\begin{matrix} n \\ 1 \end{matrix} \right]$ [(optionD [])]</p> </div> <p>OptionA: $\left[\begin{matrix} WTM \\ NOWTM \end{matrix} \right]$ $\left[\begin{matrix} BLKsize 800 \\ BLKsize 4096 \\ BLKsize 4K \\ BLKsize 32K \\ BLKsize 64K \end{matrix} \right]$ OptionB: $\left[\begin{matrix} NOPrint \\ PPrint \\ Term \\ DISK \end{matrix} \right]$</p> <p>OptionC: $\left[\begin{matrix} EOT \\ EOF n \\ EOF 1 \end{matrix} \right]$ OptionD: $\left[\begin{matrix} TAPn \\ TAP1 \\ vdev \\ 181 \end{matrix} \right]$ $\left[\begin{matrix} 7TRACK \\ 9TRACK \\ 18TRACK \end{matrix} \right]$ [DEN <i>den</i>] [TRTCH <i>a</i>]</p> <p>OptionE: $\left[\begin{matrix} REWIND \\ LEAVE \end{matrix} \right]$ OptionF: $\left[\begin{matrix} TRANSfer BUFF \\ TRANSfer IMMED \end{matrix} \right]$</p>
-------------	--

Operands

DUMP *fn ft [fm]*

.dumps one or more files to tape. If *fn* and/or *ft* is specified as an asterisk (*) all files that satisfy the other file identifier are dumped.

If *fm* is coded as a letter, that disk or directory and its extensions are searched for the specified file(s). If *fm* is coded as a letter and number, only files with that mode number and letter (and the extensions of the disk or directory

referenced by that fm letter) are dumped. If fm is coded as asterisk (*), all accessed disks and directories are searched for the specified file(s). If fm is not specified, only the disk or directory accessed as A and its extensions are searched.

LOAD [*fn ft [fm]*]

reads tape files onto a disk or directory. If a file identifier is specified, only that one file is loaded. If the option EOF n is specified and no file identifier is entered, n tape files are written to a disk or directory. If an asterisk (*) is specified for fn or ft, all files within EOF n that satisfy the other file identifier are loaded.

The files are written to the disk or directory indicated by the file mode letter. The file mode number, if entered, indicates that only files with that file mode number are to be loaded. The default file mode letter is A.

SCAN [*fn ft*]

positions the tape at a specified point, and lists the identifiers of the files it scans. Scanning occurs over n tape marks, as specified by the option EOF n (the default is 1 tape file). However, if a file identifier (fn and ft) is specified, scanning stops upon encountering that file; the tape remains positioned ahead of the file. Asterisks (*) can be specified for either fn or ft or both.

SKIP [*fn ft*]

positions the tape at a specified point and lists the identifiers of the files it skips. Skipping occurs over n tape marks, as specified by the option EOF n (the default is 1 tape mark). However, if a file identifier (fn and ft) is specified, skipping stops after encountering that file; the tape remains positioned immediately following the file. Asterisks (*) can be specified for either fn or ft or both.

MODESET

sets the values specified by the DEN, TRACK, and TRTCH options if the tape is at load point. After initial specification in a TAPE command, these values remain in effect for the virtual tape device until they are changed in a subsequent TAPE command, RDTAPE, WRTAPE, or TABECTL macros. If you do not specify a density (DEN) on the next TAPE command, or a MODESET on the next TAPE macro, the tape drive remains at the density you previously set unless a re-ipl is done.

For dual density drives, the default density is the highest density of the drive. See Usage Note 7 for further considerations.

Note: Before you use the MODESET option, you should rewind your tape to ensure that the tape is at loadpoint. Entering the command TAPE REW, positions the tape at load point.

tapcmd n

tapcmd [**1**]

specifies a tape control function (tapcmd) to be executed n times (default is 1 if n is not specified). These functions also work on tapes in a non-CMS format.

Tapcmd	Action
BSF	Backspace <i>n</i> tape marks
BSR	Backspace <i>n</i> tape records
ERG	Erase a defective section of the tape
FSF	Forward-space <i>n</i> tape marks
FSR	Forward-space <i>n</i> tape records
REW	Rewind tape to load point
RUN	Rewind tape and unload

WTM Write *n* tape marks

DVOLI

displays an 80-character VOL1 label in EBCDIC on the user's terminal if such a label exists on the tape. If the first record on the tape is not a VOL1 label, an error message is sent to the user.

WVOL1 *valid {owner}*

writes a VOL1 label on a tape. All fields are set to the same values they are set to when a VOL1 label is written by the IBM-supplied IEHINITT utility program (see the publication *OS/VS2 MVS Utilities* for details). The volume ID is set to the 1- to 6-character volume ID specified on the command. If the user specifies owner field, it is written in the owner name and address code field of the label. It can be up to eight characters long and left-justified in the 10-byte field in the label. If not specified, the owner field is set to blanks. The WVOL1 option also writes a dummy HDR1 label and tape mark after the VOL1 label.

Note: The default option of LEAVE positions the tape at the record immediately after the VOL1 label. Refer to the REWIND and LEAVE options for more information.

Options

If conflicting options are specified, the last one entered is in effect.

WTM

writes two tape marks after each file that is dumped, then backspaces over both of the tape marks so that subsequent files written on the tape are not separated by tape marks.

NOWTM

does not separate files with tape marks when multiple files are dumped. After the last file, two tape marks are written to indicate the end-of-tape (EOT). Subsequent files write over both tape marks. NOWTM is the default.

When a single file is dumped, two tape marks are written after the file. Subsequent files write over both tape marks.

BLKsize 800

BLKsize 4K

Blksize 4096

BLKsize 32K

BLKsize 64K

specifies the size of the tape data block at which the files are to be dumped (not including a five-byte prefix). The default is BLKSIZE 4096 (or 4K). For more details on tape data block sizes, see Usage Note 1.

The BLKSIZE 32K and BLKSIZE 64K options conflict with options that set the mode for a 7-track tape drive, such as 7TRACK, DEN 200, DEN 556, and TRTCH.

NOPrint

does not spool the list of files dumped, loaded, scanned, or skipped to the printer.

Print

spools the list of files dumped, loaded, scanned, or skipped to the printer.

Term

displays a list of files dumped, loaded, scanned, or skipped at the terminal.

DISK

creates a file containing the list of files dumped, loaded, scanned, or skipped. The file has the file identification of TAPE MAP A5.

EOT

reads the tape until an end-of-tape indication is received.

EOF *n***EOF 1**

reads the tape through a maximum of *n* tape marks. The default is EOF 1.

TAP*n****vdev***

specifies the symbolic tape identification (TAP*n*) or the virtual device address (*vdev*) of the tape to be read from or written to. The following symbolic names and virtual device addresses are supported:

Symbolic Name	Virtual Address	Symbolic Name	Virtual Address
TAP0	180	TAP8	288
TAP1	181	TAP9	289
TAP2	182	TAPA	28A
TAP3	183	TAPB	28B
TAP4	184	TAPC	28C
TAP5	185	TAPD	28D
TAP6	186	TAPE	28E
TAP7	187	TAPF	28F

The default is TAP1 or 181. The unit specified by *vdev* must previously have been attached to your CMS virtual machine before any tape I/O operation can be attempted.

7TRACK

specifies a 7-track tape. Odd parity, data convert on, and translate off are assumed unless TRTCH is specified.

The 7TRACK option conflicts with the BLKSIZE 32K and BLKSIZE 64K options.

9TRACK

specifies a 9-track tape.

18TRACK

specifies an 18-track tape.

DEN *den*

is the tape density where *den* is 200, 556, 800, 1600, 6250, or 38K. If 200 or 556 is specified, 7TRACK is assumed. If 1600 or 6250 is specified, 9TRACK is assumed; if 800 is specified, 9TRACK is assumed unless 7TRACK is specified. If 38K is specified, 18TRACK is assumed. In the case of either 800/1600 or 1600/6250 dual-density drives, highest density is the default if the 9TRACK option is specified. If neither the 9TRACK option nor the DEN option is specified, the drive operates at whatever BPI the tape drive was last set or the default density if none was set. The following densities are allowed for the given track sizes.

7track 200, 556, 800

9track 800, 1600, 6250

18track 38K

The DEN 200 and DEN 556 options conflict with the BLKSIZE 32K and BLKSIZE 64K options.

TRTCH a

is the tape recording technique for 7-track tape. If TRTCH is specified, 7TRACK is assumed. One of the following must be specified as "a":

a Meaning

- O Odd parity, data convert off, translate off
- OC Odd parity, data convert on, translate off
- OT Odd parity, data convert off, translate on
- E Even parity, data convert off, translate off
- ET Even parity, data convert off, translate on

The TRTCH options conflict with the BLKSIZE 32K and BLKSIZE 64K options.

REWIND

LEAVE

are only valid for the DVOL1 and WVOL1 functions. They specify the positioning of a tape after the VOL1 is processed. If REWIND is specified, the tape is rewound and positioned at load point. If LEAVE (the default) is specified, the tape is positioned at the record immediately after the VOL1 label.

TRANSfer BUFF

TRANSfer IMMED

Specifies the tape write mode for the 3480 Magnetic Tape Subsystem. The two write modes are:

- Buffered Write Mode (BUFF)
- Tape Write Immediate Mode (IMMED).

BUFF mode is the default

In BUFF mode, data is transferred from the processor to the tape control unit, then the processor and tape control unit are disconnected from each other. The tape control unit then writes the data onto the tape and performs error checking and, if necessary, error recovery procedures.

In IMMED mode, data is physically written onto tape and "read-back" checked (verified) by the microprogram in the control unit while the processor and the control unit are still connected. This mode is provided, at a severe performance degradation, for nonrestartable write operations.

Usage Notes

1. There are four types of tape records written by the TAPE command. The formats are determined by the BLKSIZE option.

If you use the BLKSIZE 800 option, the tape records are 805 bytes long. If the BLKSIZE option is specified as, or defaults to, 4096 (or 4K), the tape records are 4101 bytes long. The first byte in the tape header is a binary 2 (X'02'). The next three bytes contain CMS, followed by a file format byte. For a variable format file, the file format byte is V, for a fixed format file the file format byte is F, and for a fixed format sparse file, the file format byte is S.

In the final record, the character N replaces the file format byte, and the data area contains CMS file directory information. A tape created at 4096-byte blocksize is not reloadable on a CMS system that does not have the multivalued

BLKSIZE option on the TAPE command; however, the 800-byte BLKSIZE option provides backward compatibility to such a system.

If you use the BLKSIZE 32K option, the tape records are 32,767 (32K-1) bytes long. If the BLKSIZE option is specified as 64K, the tape records are 65,535 (64K-1) bytes long. The BLKSIZE 32K and BLKSIZE 64K options optimize the performance of the tape drive. The first byte in the tape header is one of the following:

- X'10'** 32K tape record contains data only.
- X'50'** This last (and possibly only) 32K tape record contains file directory information.
- X'20'** 64K tape record contains data only.
- X'60'** This last (and possibly only) 64K tape record contains file directory information.

The next three bytes contain CMS, followed by a file format byte. For a variable format file, the file format byte is V, for a fixed format file, the file format byte is F, and for a fixed format sparse file, the file format byte is S. In the final (or only) record, the data area contains CMS file directory information and may also contain file data.

The BLKSIZE 32K and BLKSIZE 64K options are not supported on 7-track tape drives.

2. If a tape file contains more CMS files than would fit on a disk, the tape load operation may terminate if there is not enough disk space to hold the files. Similarly, in the Shared File System, if the directory to which you are writing runs out of file space, the tape load operation may terminate. To prevent this, when you dump the files, separate the logical files by tape marks, then forward space to the appropriate file.
3. Because the CMS file directory is the last record of the file, the TAPE command creates a separate workfile so that backspacing and rereading can be avoided when the file is built. If the load criteria is not satisfied, the workfile is erased; if it is satisfied, the workfile is renamed. This workfile is named TAPE CMSUT1, which may exist if a previous TAPE command has abnormally terminated. If the work file is accidentally dumped to tape and subsequently loaded, it appears on your disk or directory as TAPE CMSUT2.
4. The RUN option (rewind and unload) indicates completion before the physical operation is completed. Thus, a subsequent operation to the same physical device may encounter a device busy situation.
5. It is possible to run a tape off the reel in at least one situation. If you specify EOF *n* and *n* is greater than the number of tape marks on the tape, the tape will run off the reel.
6. The DVOL1 and WVOL1 are the only TAPE command functions that automatically process tape labels. TAPE DUMP does not automatically write labels on a tape when it writes the dump file, and TAPE LOAD does not recognize tape labels when loading a file.

TAPE

7. To reset the mode of a variable-density tape drive when using an IBM standard label tape for output, rewrite the VOL1 label before processing tape using the TAPE WVOL1 command. This is a hardware restriction which allows changes to the tape drive mode only when the tape is at load point. If you are writing to a non-label tape, use the TAPE MODESET command to set the mode. The first write operation will cause the mode to be reset since the tape will be at loadpoint when the write takes place.

If you are switching from one drive to another on the same device, message DMSTPJ110S (DMSTPJ111S) may be received because the current device does not support the density contained in the device table (DEVTABA). The TAPE command must be issued with the desired modeset to eliminate the DMSTPJ110S (DMSTPJ111S) error message. Regarding DENSITY (MODE), if the mode is omitted by the user, the mode is taken from the DEVTAB in the nucleus. If there is no mode stored in the device table, the mode is set to the highest density of the drive.

8. Do not use TAPE DVOL1 for a tape that you suspect to be blank. If you do, and the tape is blank, it will run off the reel.
9. The options for the 8809 and 9347 tape drive must be 9TRACK and DEN 1600. Note that these are the default values, so you do not need to specify them.
10. For more information on tape file handling, see the *VM/SP CMS User's Guide*. Tapes with a density of 38K BPI and 18TRACK are used only by the 3480 Magnetic Tape Subsystem. This subsystem does not read or write current half-inch tape (such as used by the 2400 and other 3400 subsystems). Data on a 3480 cartridge must be copied onto a half-inch tape to interchange with systems that do not have a 3480 subsystem.
11. The first time that the TAPE command is issued, it is invoked from the transient area; thereafter it is invoked as a nucleus extension.
12. If you specify either TAPE or VMFPLC2 as a synonym of the other, do not use the synonym to call that function from within an exec. You may use any name other than TAPE or VMFPLC2 as a synonym of the other function. For example, from within an exec, TAPE is not a valid synonym for VMFPLC2; TAP however, would be valid.

Responses

DMSTPF701I Null file

A final record was encountered and no prior records were read in a TAPE LOAD operation. No file is created.

If the TERM option is in effect, the following is displayed at the terminal depending on the operation specified:

```

Loading ...
  fn ft fm
  . . .
  . . .
  . . .

```

```

Skipping ...
  fn ft fm
  . . .
  . . .
  . . .

```

```

Dumping ...
  fn ft fm
  . . .
  . . .
  . . .

```

```

Scanning ...
  fn ft fm
  . . .
  . . .
  . . .

```

When a tape mark is encountered, the following is displayed at the terminal if the TERM option is specified:

End-of-file or end-of-tape

Messages and Return Codes

DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC = 31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSLMX264E	<i>command</i> is not a valid command to be established as a nucleus extension by DMSLMX [RC = 24]
DMSLMX514E	Return code <i>nnn</i> from NUCXLOAD
DMSTPF701I	Null file
DMSTPI613E	Tape must be invoked as a nucleus extension [RC = 40]
DMSTPJ002E	File(s) [<i>fn</i> [<i>ft</i> [<i>fm</i>]]] not found [RC = 28]
DMSTPJ003E	Invalid option: <i>option</i> [RC = 24]
DMSTPJ010E	Premature EOF on file [(<i>fn ft</i> [<i>fm</i>] number <i>nn</i>)] [RC = 40]
DMSTPJ014E	Invalid function <i>function</i> [RC = 24]
DMSTPJ017E	Invalid device address <i>vdev</i> [RC = 24]
DMSTPJ023E	No filetype specified [RC = 24]
DMSTPJ027E	Invalid device <i>devtype</i> [for SYSaaa] [RC = 24]
DMSTPJ029E	Invalid parameter <i>parameter</i> in the option <i>option</i> field [RC = 24]
DMSTPJ037E	Filemode <i>mode</i> is read/only [RC = 36]
DMSTPJ042E	No fileid(s) specified [RC = 24]
DMSTPJ043E	Tapn(<i>vdev</i>) is file protected [RC = 36]
DMSTPJ047E	No function specified [RC = 24]
DMSTPJ048E	Invalid mode <i>mode</i> [RC = 24]
DMSTPJ057E	Invalid record format [RC = 32]
DMSTPJ058E	End-of-file or end-of-tape [RC = 40]
DMSTPJ069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSTPJ070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSTPJ096E	File <i>fn ft</i> data block count incorrect [RC = 32]
DMSTPJ104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]
DMSTPJ105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSTPJ109S	Virtual storage capacity exceeded [RC = 104]

DMSTPJ110S Error reading *tapn(vdev)* [RC = 100]
 DMSTPJ111S Error writing *tapn(vdev)* [RC = 100]
 DMSTPJ113S *Tapn(vdev)* not attached [RC = 100]
 DMSTPJ115S Conversion feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S Dual density feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S Translation feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 7-track feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 9-track feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 18-track feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 800 BPI feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 1600 BPI feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 6250 BPI feature not supported on device *vdev* [RC = 88]
 DMSTPJ115S 32K blocksize not supported on device *vdev* [RC = 88]
 DMSTPJ115S 64K blocksize not supported on device *vdev* [RC = 88]
 DMSTPJ335W *Tapn [(vdev)]* has been manually rewound and unloaded.
 Requested tape function may not have been executed. [RC = 4]
 DMSTPJ431E *Tapn(vdev)* VOL1 label missing [RC = 32]
 DMSTPJ671E Error loading file *fn ft fm; rc = nn* from RENAME [RC = 100]
 DMSTPJ671E Error loading file *fn ft fm rc = nn* from COPYFILE [RC = 100]
 DMSTPJ1262S Error *nn* opening file *fn ft fm|dirid* [RC = 40]
 DMSTPJ1262S Error *nn* opening file *fn ft fm|dirid* [RC = 104]
 DMSTPJ1262S Error *nn* closing file *fn ft fm|dirid* [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in using a file	814
Errors in copying a file	70
Errors in erasing a file	145
Errors in renaming a file	491

TAPEMAC

Use the TAPEMAC command to create a CMS MACLIB from an unloaded partitioned data set (PDS) from a tape created by the IEHMOVE utility program under OS. The PDS from which the tape was created can be blocked, but the logical record length must be 80.

Format

TAPEMAC	$fn \left[\begin{array}{l} \underline{\text{SL}} \quad [labeldefid] \\ \underline{\text{NSL}} \quad filename \quad [ID = identifier] \end{array} \right] [(\text{options... } ())]$ <p style="text-align: center;"> <u>Options:</u> $\left[\begin{array}{l} \underline{\text{TAP}n} \\ \underline{\text{TAP}1} \end{array} \right]$ $\left[\begin{array}{l} \underline{\text{ITEMCT}} \quad yyyyyy \\ \underline{\text{ITEMCT}} \quad 50000 \end{array} \right]$ </p>
----------------	--

Operands

fn

specifies the file name of the first, or only, CMS MACLIB to be created on the disk or directory accessed as A. If *fn* MACLIB already exists on the disk or directory accessed as A, the old one is erased; no warning message is issued.

SL

means that the tape has standard labels. The default is SL without a *labeldefid*. With the default specification, the standard header labels are only displayed on the user's terminal. If *labeldefid* is specified, the standard labels are not displayed, but are checked by the tape label checking routine.

NSL

means that the tape has nonstandard labels.

labeldefid

identifies the LABELDEF command that supplies descriptive label information for the file to be processed. The *labeldefid* given here must match the 1- to 8-character identifier specified as the file name on the LABELDEF command that was previously issued.

filename

is the CMS file name of a routine to process nonstandard labels. The file type must be TEXT or MODULE. If both TEXT and MODULE files exist, the MODULE file is used. MODULE files that are used for NSL routines with the TAPEMAC command must be created so that they start at an address above X'21000'. This prevents the NSL modules from overlaying the command. See the section "Tape Labels in CMS" in the *VM/SP CMS User's Guide* for details on how to write routines to process nonstandard labels.

ID = identifier

specifies a 1- to 8-character identifier to be passed to a user-written NSL routine. You may use the identifier in any way you want to identify the file being processed. The identifier is passed to the user routine exactly as specified in the ID operand. If an identifier is not specified, blanks are passed. See the section

“Tape Labels in CMS” in the *VM/SP CMS User’s Guide* for details on communicating with routines that process nonstandard labels.

Options

TAPn

specifies the symbolic name of the tape. The following names are supported and represent these virtual units:

Symbolic Name	Virtual Address	Symbolic Name	Virtual Address
TAP0	180	TAP8	288
TAP1	181	TAP9	289
TAP2	182	TAPA	28A
TAP3	183	TAPB	28B
TAP4	184	TAPC	28C
TAP5	185	TAPD	28D
TAP6	186	TAPE	28E
TAP7	187	TAPF	28F

The default is TAP1.

ITEMCT yyyyy

specifies the item count threshold of each MACLIB to be created, which is the maximum number of records to be written into each file. Commas are not allowed. If ITEMCT is not specified, the default is 50000.

Usage Notes

1. Tape records are read and placed into *fn* MACLIB until the file size exceeds the ITEMCT (item count); loading then continues until the end of the current member is reached. Then another CMS file is created; its file name consists of the number 2 appended to the end of the file name specified (*fn*) if the file name is seven characters or less. The appended number overlays the last character of the file name if the name is eight characters long. Loading then continues with this new name. For example, if you enter the command:

```
tapemac mylib
```

you may create files named MYLIB MACLIB, MYLIB2 MACLIB, MYLIB3 MACLIB, and so on.

This process continues until up to nine CMS files have been created. If more data exists on the tape than can fit in nine CMS files, processing is terminated with the error message DMSTMA139S. A maclib created by the TAPEMAC command may contain a maximum of 256 MACLIB directory entries.

2. Only header labels of the first file encountered are displayed or checked if SL or SL labdefid is specified. Trailer labels are not processed or displayed; they are skipped.
3. The following examples illustrate the different ways tape labels are processed by TAPEMAC. The command

```
tapemac mac6 sl
```

displays any standard VOL1 or HDR1 labels on a tape before loading maclib MAC6. It does not stop before loading the MACLIB.

If you specify

```
labeldef taplab fid macfile crdte 77106
tapemac mac8 sl taplab
```

CMS checks the HDR1 label on the tape before loading MAC8. It uses the information you supplied in the LABELDEF command TAPLAB to check the label. If there are discrepancies between fields you specified in the LABELDEF command and in the actual tape label, the MACLIB is not loaded.

If you specify

```
tapemac mac10 ns1 ns13
```

CMS uses your own routine NSL3 to process tape labels before loading MAC10.

Responses

The TAPEMAC command displays the message:

```
LOADING fn MACLIB
```

for each macro library created.

Messages and Return Codes

DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC=31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSTMA001E	No filename specified [RC=24]
DMSTMA003E	Invalid option: <i>option</i> [RC=24]
DMSTMA027E	Invalid device <i>devtype</i> [for SYSaaa] [RC=24]
DMSTMA057E	Invalid record format [RC=32]
DMSTMA069E	Filemode <i>mode</i> not accessed [RC=36]
DMSTMA070E	Invalid parameter <i>parameter</i> [RC=24]
DMSTMA105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC=100]
DMSTMA109S	Virtual storage capacity exceeded [RC=104]
DMSTMA110S	Error reading <i>tapn(vdev)</i> [RC=100]
DMSTMA138S	Error <i>nn</i> erasing <i>fn ft</i> before loading tape [RC=100]
DMSTMA139S	Tape file exceeds 9 CMS MACLIBs [RC=104]
DMSTMA335W	<i>Tapn [(vdev)]</i> has been manually rewound and unloaded. Requested tape function may not have been executed. [RC=4]
DMSTMA420E	NSL exit filename missing or invalid [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

TAPPDS

Use the TAPPDS command to create CMS files from tapes that are used as input to or output from the following OS utility programs:

- IEBPTPCH -- tape files must be the result of an IEBPTPCH punch operation from either a sequential or partitioned data set in OS. The default attributes (IEBPTPCH DCB) must have been issued:

```
DCB=(RECFM=FA,LRECL=81,BLKSIZE=81)
```

- IEBUPDTE -- tape files may be blocked or unblocked and must be in the format accepted by IEBUPDTE as "control data set" (SYSIN) input with a control statement

```
./ ADD...
```

preceding the records to be placed in each partitioned data set member (OS) or separate CMS file (CMS)).

- IEHMOVE -- unloaded partitioned data sets are read.

The tape can contain OS standard labels or be unlabeled.

Format

TAPPDS	$\left[\begin{array}{c} fn \\ * \end{array} \left[\begin{array}{c} ft \\ * \end{array} \left[\begin{array}{c} fm \\ A1 \\ * \end{array} \right] \right] \right] \left[\begin{array}{l} \underline{SL} \ [labeldefid] \\ \underline{NSL} \ filename \ [ID = identifier] \end{array} \right] \left[(\text{options...}[]) \right]$ <p>Options: $\left[\begin{array}{l} \underline{PDS} \\ \underline{NOPDS} \\ \underline{UPDATE} \end{array} \right] \left[\begin{array}{l} \underline{COL1} \\ \underline{NOCOL1} \end{array} \right] \left[\begin{array}{l} \underline{TAPn} \\ \underline{TAP1} \end{array} \right] \left[\begin{array}{l} \underline{END} \\ \underline{NOEND} \end{array} \right] \left[\begin{array}{l} \underline{MAXTEN} \\ \underline{NOMAXTEN} \end{array} \right]$</p>
---------------	--

Operands

fn

is the file name of the file to be created from the sequential tape file. If the tape contains members of a partitioned data set (PDS), *fn* must be specified as an asterisk (*); one file is created for each member with a file name the same as the member name. If NOPDS or UPDATE is specified and you do not specify *fn* or specify it as an asterisk (*), the default file name is TAPPDS.

ft

is the file type of the newly created files. The default file types are CMSUT1 (for PDS or NOPDS) and ASSEMBLE (for UPDATE). The defaults are used if *ft* is omitted or specified as *.

fm

is the file mode of the disk or directory to contain the new files. If this field is omitted or specified as an asterisk (*), A1 is assumed.

SL

means that the tape has standard labels. The default is SL without a `labeldefid`. With the default specification, the standard labels are displayed at the user's terminal. If `labeldefid` is specified, the standard labels are not displayed, but are checked by the tape label checking routine.

NSL

means that the tape has nonstandard labels.

labeldefid

identifies the LABELDEF command, which supplies descriptive label information for the file to be processed. The `labeldefid` given here must match the 1- to 8-character specified as the file name on the LABELDEF command that was previously issued.

filename

is the CMS file name of a routine to process nonstandard labels. The file type must be TEXT or MODULE. If both TEXT and MODULE files exist, the MODULE file is used. MODULE files that are used for NSL routines with the TAPPDS command must be created so that they start at an address above X'21000'. This prevents the MODULE files from overlaying the command. See the section "Tape Labels in CMS" in the *VM/SP CMS User's Guide* for details on writing routines to process nonstandard labels.

ID = identifier

specifies a 1- to 8-character identifier to a user-written NSL routine. You may use the identifier in any way you want to identify the file being processed. The identifier is passed to the user routine exactly as specified in the operand. If an identifier is not specified, blanks are passed. See the section "Tape Labels in CMS" in the *VM/SP CMS User's Guide* for details on communication with routines that process nonstandard labels.

Note: If either SL or NSL is specified for tape label processing, the `fn`, `ft`, and `fm` operands must all be specified. They may be specified by asterisks (*) if you want default values; however, none of the three operands may be omitted.

Options

If conflicting options are specified, the last one entered is the one that is used. All options, except TAPn, are ignored when unloaded (IEHMOVE) PDS tapes are read.

PDS

indicates that the tape contains members of an OS partitioned data set, each preceded by a MEMBER NAME = name statement. The tape must have been created by the OS IEBTPCH service program if this option is specified.

NOPDS

indicates that the contents of the tape will be placed in one CMS file.

UPDATE

indicates that the tape file is in IEBUPDTE control file format. The file name of each file is taken from the NAME = parameter in the "/ ADD" record that precedes each member. (See Usage Note 2.)

COL1

reads data from columns 1-80. You should specify this option when you use the UPDATE option.

NOCOL1

reads data from columns 2-81; column 1 contains control character information. This is the format produced by the OS IEBTPCH service program.

TAP_n

specifies the symbolic name of the tape device. The following names are supported and represent these virtual units:

Symbolic Name	Virtual Address	Symbolic Name	Virtual Address
TAP0	180	TAP8	288
TAP1	181	TAP9	289
TAP2	182	TAPA	28A
TAP3	183	TAPB	28B
TAP4	184	TAPC	28C
TAP5	185	TAPD	28D
TAP6	186	TAPE	28E
TAP7	187	TAPF	28F

If not specified, TAP1 is assumed.

END

considers an END statement (characters 'END' in columns 2-5) a delimiter for the current member.

NOEND

specifies that END statements are not to be treated as member delimiters, but are to be processed as text.

MAXTEN

reads up to ten members. This is valid only if the PDS option is selected.

NOMAXTEN

reads any number of members.

Usage Notes

1. You can use the TAPE command to position a tape at a particular tape file before reading it with the TAPPDS command. If the tape has OS standard labels, TAPDDS will read and display the "VOL1" and "HDR" records at the terminal. If the file you want to process is not at the beginning of the tape, the TAPE command must be used to position the tape at a particular tape file before reading it with the TAPPDS command. Be aware that each file on an OS standard label tape is actually three physical files (HDR, DATA, TRAILER). If positioning to other than the first file, the user must skip more physical tape files (3n-3 if positioning to the header labels, 3n-2 if positioning to the data file, where n is the number of the file on the tape).
2. If you use the UPDATE option, you must also specify the COL1 option. Each tape record is scanned for a "./ ADD" record beginning in column 1. When a "./ ADD" record is found, subsequent records are read onto disk (or directory) until the next "./ ADD" record is encountered or until a "./ ENDUP" record is encountered.

A "./ ENDUP" record or a tape mark ends the TAPPDS command execution; the tape is not repositioned.

"/ label" records are not recognized by CMS and are included in the file as data records.

If the NAME= parameter is missing on the "./ ADD" record or if it is followed by a blank, TAPPDS uses the default file name, TAPPDS, for the CMS file. If

this happens more than once during the execution of the command, only the last unnamed member is contained in the TAPPDS file.

3. If you are reading a macro library from a tape created by the IEHMOVE utility, you can create a CMS MACLIB file directly by using the TAPEMAC command.
4. Only header labels of the first file encountered are displayed or checked if SL or SL labeldefid is specified. Trailer labels are not processed or displayed; they are skipped. When more than one file is processed by one issuance of the TAPPDS command, only the first file has its standard labels processed. Standard labels are skipped on succeeding files.
5. The following examples illustrate different ways in which tape labels are processed by TAPPDS. If you specify:

```
tappds fileg cmsut1 * sl
```

then, before loading the PDS into fileg, CMS displays a VOL1 and HDR1 label if it exists on the tape. It does not stop before the PDS is loaded; therefore, you cannot use the tape label to suppress loading if the wrong tape has been mounted.

If you specify:

```
labeldef label2 fid pds1 volid xyz
tappds fileh cmsut1 * sl label2
```

CMS uses the label information specified to check the label on the tape before loading your PDS. If there are discrepancies, the PDS is not loaded.

If you specify

```
tappds filej * * nsl nonstd
```

CMS uses your own routine called NONSTD to process tape labels before loading the PDS.

Responses

```
DMSTPD703I   File fn ft [fm] copied
```

The named file is copied to a disk or directory.

```
DMSTPD707I   Ten files copied
```

The MAXTEN option was specified and ten members have been copied.

Note: If the tape being read contains standard OS labels, the labels are displayed at the terminal.

Messages and Return Codes

```
DMSTPD003E   Invalid option: option [RC = 24]
DMSTPD027E   Invalid device devtype [for SYSaaa] [RC = 24]
DMSTPD058E   End-of-file or end-of-tape [RC = 40]
DMSTPD105S   Error nn writing file fn ft fm on disk or directory [RC = 100]
DMSTPD109S   Virtual storage capacity exceeded [RC = 104]
DMSTPD110S   Error reading tapn(vdev) [RC = 100]
DMSTPD335W   Tapn [(vdev)] has been manually rewound and unloaded.
              Requested tape function may not have been executed. [RC = 4]
DMSTPD420E   NSL exit filename missing or invalid [RC = 24]
```

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in the Shared File System	813
Errors in using a file	814

TELL

Use the TELL command to send a message to one or more computer users on your computer or on other computers that are connected to yours via the Remote Spooling Communications Subsystem (RSCS) network. These users must be logged on to receive your message.

TELL is one of several commands that references a “userid NAMES” file. By setting up a names file, you can identify recipients just by using nicknames, which are automatically converted into node and user ID. For information on creating a NAMES file, see the NAMES command.

Format

TELL	<i>name message</i>
-------------	---------------------

Operands*name*

is the “name” of the computer user to whom the message is to be sent. If the same recipient is specified more than once, he receives only one message. The “name” may take any of the following forms:

- A “nickname” that can be found in the file “userid NAMES,” where “userid” is your user ID. This nickname may represent a single person (on your computer or on another computer), or a list of people. If the nickname represents a list, the message is sent to everyone on the list.
- A user ID of a user on your computer. If a name cannot be found in the “userid NAMES” file, it is assumed to be a user ID of someone on your computer.
- “userid AT node” which identifies a user (“userid”) on your computer or another computer (“node”). The “userid NAMES” file is not examined in this case.

You cannot send messages to a user ID named “AT” or “CC:.”

message

is the text of the message that is sent.

Usage Notes

1. If the first word of your message is “at,” you must use the third form of “name” (shown above).
2. If the person to whom you are sending the message either is not logged on or is not accepting messages (by issuing CP SET MSG OFF), he will not receive the message.
3. The TELL command uses the CP MESSAGE command to send messages to users logged on to your computer. If you would like to send messages using a different CP command (specifically, MSGNOH, SMSG, or WNG) and you are an authorized user, you change the command that TELL uses with the DEFAULTS command. See the description of the DEFAULTS command in this publication. See the description of the MSGNOH, SMSG, and WARNING commands in the *VM/SP CP General User Command Reference*.

TELL

4. The TELL command uses the CP SMSG command to send messages, via RSCS, to users logged on to other computers (nodes). A warning message may occur if the SMSG command created by TELL is too long to be handled by RSCS. In this case, shorten the message text or split it into shorter messages, and then reissue the TELL command.
5. If you want to issue TELL from an exec program, you should precede it with the EXEC command; that is, specify
exec tell

Messages and Return Codes

- DMSWTL399E Too many tags or tag too long for *nickname* in *userid* NAMES file [RC = 88]
DMSWTL499E User not authorized to issue *command* command [RC = 40]
DMSWTL647E Userid not specified for *nickname* in *userid* NAMES file [RC = 32]
DMSWTL648E Userid *name* not found; no message has been sent [RC = 32]
DMSWTL676E Invalid character * for Network ID [RC = 20]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

TXTLIB

Use the TXTLIB command to update CMS text libraries.

Format

TXTlib	<pre> { GEN libname fn1 [fn2...] [(optionA ())] ADD libname fn1 [fn2...] [(optionA ())] DEL libname membername1 [membername2...] MAP libname [(optionB ())] } OptionA: [FILENAME] OptionB: [TERM DISK PRINT] </pre>
---------------	--

Operands

GEN

creates a TXTLIB on your disk or directory accessed as A. If a TXTLIB with the same name already exists, it is replaced.

ADD

adds TEXT files to the end of an existing TXTLIB on a read/write disk or directory. No checking is done for duplicate names, entry points, or CSECTs.

DEL

deletes members from a TXTLIB on a read/write disk or directory and compresses the TXTLIB to remove unused space. If more than one member exists with the same name, only the first entry is deleted.

MAP

lists the names (entry points) of TXTLIB members, their locations in the library, and the number of entries.

libname

specifies the file name of a file with a file type of TXTLIB, which is to be created or listed or from which members are to be deleted or added.

fn1 [fn2...]

specifies the name(s) of file(s) with file type(s) of TEXT, that you want to add to a TXTLIB.

membername1 [membername2...]

specifies the name(s) of TXTLIB member(s) that you want to delete.

TXTLIB

Option A

FILENAME

indicates that all the file names specified will be used as the member names for their respective entries in the TXTLIB file instead of the first CSECT in the file's text deck.

Option B

TERM

displays information about the TXTLIB on your terminal.

DISK

writes a CMS file, named "libname MAP A5," that contains a list of TXTLIB members. If a file of that name already exists, the old file is erased. The DISK option is the default.

PRINT

spools a copy of the TXTLIB map to the virtual printer.

Usage Notes

1. The FILENAME option overrides any name card in a text file. The name card functions as before, but the specified file name becomes the member name in the TXTLIB. The name card is the only entry within that member name of the TXTLIB. If a name card is not found in the text file and you specify the FILENAME option, the file's name is the member name. The first CSECT in the text file is the first entry point (the remaining entry points in the text file follow) within that member.
2. The FILENAME option sets the member name to the file name and makes the first CSECT the first entry point. Therefore, any text file that has 255 entry points will exceed this limit by one if you select the FILENAME option. In other words, if you select the FILENAME option, the maximum number of entry points a file may contain is 254.
3. When a TEXT file is added to a library, its member name is taken from the first CSECT name, or, if a NAME card exists, from the NAME statement in the TEXT file. All other entry points in the TEXT file become entries within this member unless there is a NAME card. In this case, the only entry created is the member name. For example, a TEXT file with a file name of TESTPROG that contains CSECTs named CHECK and RECHECK, when added to a TXTLIB, creates a member named CHECK and an entry point named RECHECK within this member. If it contained a NAME statement at the end of the text deck, that name would be the only entry created for that text in the TXTLIB. Deletions and LOAD and INCLUDE command references must be made on the member name.
4. If you create an alias for a TXTLIB, using the CREATE ALIAS command, the alias must have a file type of TXTLIB.
5. Members must be deleted by their initial entry in the dictionary (that is, their "name" or the first ID name). Any attempt to delete a specific alias or entry point within a member will result in a "Not found" message.
6. If you want your TXTLIBs to be searched for missing subroutines during CMS loader processing; you must identify the TXTLIB on a GLOBAL command; for example:

```
global txtlib newlib
```

7. You may add OS linkage editor control statements NAME, ALIAS, ENTRY, and SETSSI to a TEXT file before adding it to a TXTLIB. There must be a blank in column 1 and the ALIAS, NAME, and ENTRY statements must follow the END statement. The ALIAS statement must precede the NAME statement. The specified ENTRY point must be located within the CSECT.
8. TXTLIB members are not fully link-edited, and may return erroneous entry points during dynamic loading.
9. The total number of members in the TXTLIB file cannot exceed 2000.
 When this number is reached, an error message is displayed. The total number of entry points in a member cannot exceed 255. When this number is reached, an error message is displayed and the next text file (if there is one) is processed. The text library created includes all the text files entered up to (but not including) the one that caused the overflow.
10. The PRINT and TERM options erase the old MAP file, if one exists.
11. A return code of 4 indicates that an error occurred that did not terminate processing. Check the messages to determine the error. If the library is on a minidisk and the regeneration of the library or deleting from the library results in a library with no members, the library is erased. If the empty library is on an SFS directory, the library is maintained with a header record indicating that the library has no members. The empty library is maintained to preserve any file sharing authorities specified for them. The use of these empty libraries by other CMS commands, OS simulation macros, and other applications may produce unpredictable results.

Responses

When the TXTLIB MAP command is issued with the TERM option, the contents of the directory of the specified text library are displayed at the terminal. The number of entries in the text library (xxx) is also displayed.

Note: Alias names follow the main member and they do not have a location field.

```
ENTRY INDEX
name location
.
.
.
xxx ENTRIES IN LIBRARY
```

Messages and Return Codes

- DMSERD257T Internal system error at address *addr* (offset *offset*)
- DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
- DMSLBT001E No {file|CSECT} name specified [RC=24]
- DMSLBT002W File *fn ft [fm]* not found [RC=4 or 28]
- DMSLBT003E Invalid option: *option* [RC=24]
- DMSLBT013E Member *membername* not found in file *fn ft fm* [RC=32]
- DMSLBT037E Filemode *mode[(vdev)]* is accessed as read/only [RC=36]
- DMSLBT056E File *fn ft [fm]* contains invalid [name|alias|entry|ESD] record formats [RC=32]
- DMSLBT069E Filemode *mode* not accessed [RC=36]
- DMSLBT104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]
- DMSLBT105S Error *nn* writing file *fn ft fm* on disk or directory [RC=31|55|70|76|99|100]

TXTLIB

DMSLBT106S Number of member names exceeds maximum of 2000; file *fn*
TEXT not added [RC=88]
DMSLBT213W Library *fn* TXTLIB not created [RC=4]
DMSLBT213W Library *fn* TXTLIB not created, or erased if empty [RC=4]
DMSLBT213W Library *fn* TXTLIB has no members [RC=4]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813
Errors in using a file	814
Errors in copying a file	70
Errors in erasing a file	145
Errors in renaming a file	491

TYPE

Use the TYPE command to display all or part of a CMS file at the terminal in either EBCDIC or the hexadecimal representation of the EBCDIC code.

Format

Type	$fn\ ft\ \left[\begin{array}{c} fm \\ * \\ _ \end{array} \right]\ \left[\begin{array}{c} recl \\ * \\ _ \\ 1 \end{array} \right]\ \left[\begin{array}{c} recn \\ * \\ _ \end{array} \right]\ \left[(\text{options...}) \right]$ <p>options:</p> $\left[\text{HEX} \right]\ \left[\text{COL} \left\{ \begin{array}{c} xxxxx \\ 1 \end{array} \right\} - \left[\begin{array}{c} yyyyy \\ recl \end{array} \right] \right]\ \left[\text{MEMber} \left\{ \begin{array}{c} * \\ name \end{array} \right\} \right]$
-------------	---

Operands

fn
is the file name of the file to be displayed.

ft
is the file type of the file to be displayed.

fm
is the file mode of the file to be displayed. If this field is omitted, the disk or directory accessed as A and its extensions are searched to locate the file. If *fm* is specified as an asterisk (*), all disks and directories are searched, and the first file found is displayed.

recl
is the record number of the first record to be displayed. This field cannot contain special characters. If *recl* is greater than the number of records in the file, an error message is displayed. If this field is omitted or entered as an asterisk (*), a record number of 1 is assumed.

recn
is the record number of the last record to be displayed. This value cannot contain embedded commas. If this field is not specified, is entered as an asterisk (*), or is greater than the number of records in the file, displaying continues until end of file is reached.

Options

COL xxxxx-yyyyy
displays only certain columns of each record. Column xxxxx specifies the start column and yyyyy the end column of the field within the record that is to be displayed. The string xxxxx-yyyyy may have a maximum of eight characters; additional characters are truncated.

If columns are not specified, the entire record is displayed unless the file type is LISTING, in which case the first position of each record is not displayed, since it is assumed to be a carriage control character.

TYPE

HEX

displays the file in hexadecimal format.

MEMBER *

MEMBER *name*

displays member(s) of a library. If the format of the file is MACLIB or TXTLIB, a MEMBER entry can be specified. If an asterisk (*) is specified, all members of the library are displayed. If a name is specified, only that particular member is displayed. The MEMBER option should only be used with MACLIB or TXTLIB format files. With other format files, results may be unpredictable.

Usage Notes

1. If the HEX option is specified, each record can be displayed in its entirety; if not, no more than 130 characters of each record can be displayed.
2. The length of each output line is limited to 130 characters or the current terminal linesize (as specified by the CP TERMINAL command), whichever is smaller.
3. If the MEMBER option is specified more than once, only the last member specified will be typed. However, if one MEMBER option is coded with an asterisk (*), and another MEMBER option is specified with a member name, only the member specified by member name will be typed, regardless of their order on the command line.

For example, if you code:

```
TYPE ONE MACLIB (MEMBER EXAMPLE1 MEMBER EXAMPLE2
```

only EXAMPLE2 will be typed. If you code:

```
TYPE ONE MACLIB (MEMBER EXAMPLE1 MEMBER *
```

only EXAMPLE1 will be typed.

Responses

The file is displayed at the terminal according to the given specifications. When you use the HEX option, each record is preceded by a header record:

```
RECORD nnnnnnnnnn LENGTH=nnnnnnnnnn
```

Messages and Return Codes

DMSOPN002E	File [<i>fn</i> [<i>ft</i> [<i>fm</i>]]] not found [RC = 28]
DMSTYP003E	Invalid option: <i>option</i> [RC = 24]
DMSTYP005E	No <i>option</i> specified [RC = 24]
DMSTYP009E	Column <i>col</i> exceeds record length [RC = 24]
DMSTYP013E	Member <i>membername</i> not found in library [RC = 32]
DMSTYP029E	Invalid parameter <i>parameter</i> in the option <i>option</i> field [RC = 24]
DMSTYP033E	File <i>fn ft fm</i> is not a library [RC = 32]
DMSTYP039E	No entries in library <i>fn ft fm</i> [RC = 32]
DMSTYP049E	Invalid line number <i>nn</i> [RC = 24]
DMSTYP054E	Incomplete fileid specified [RC = 24]
DMSTYP062E	Invalid * in fileid [RC = 20]
DMSTYP069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSTYP104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

UPDATE

Use the UPDATE command to modify program source files. The UPDATE command accepts a source input file, and one or more files containing UPDATE control statements, updated source records, and requisite information; then it creates an updated source output file, an update log file indicating what changes, if any, were made, and an update record file if more than a single update file is applied to the input file.

Format

Update	$fn1 \left[\begin{array}{l} ft1 \\ \underline{\text{ASSEMBLE}} \end{array} \left[\begin{array}{l} fm1 \left[fn2 \left[ft2 \left[fm2 \right] \right] \right] \right] \right] \left[(\text{options...} ()) \right]$ <p>Options:</p> <table style="width: 100%; border: none;"> <tr> <td style="border: 1px solid black; padding: 2px;">[REP NOREP]</td> <td style="border: 1px solid black; padding: 2px;">[SEQ8 NOSEQ8]</td> <td style="border: 1px solid black; padding: 2px;">[INC NOINC]</td> <td style="border: 1px solid black; padding: 2px;">[CTL NOCTL]</td> <td style="padding: 2px;">[OUTMODE <i>fm</i>]</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">[STK NOSTK]</td> <td style="border: 1px solid black; padding: 2px;">[TERM NOTERM]</td> <td style="border: 1px solid black; padding: 2px;">[DISK PRINT]</td> <td style="border: 1px solid black; padding: 2px;">[STOR NOSTOR]</td> <td></td> </tr> </table>	[REP NOREP]	[SEQ8 NOSEQ8]	[INC NOINC]	[CTL NOCTL]	[OUTMODE <i>fm</i>]	[STK NOSTK]	[TERM NOTERM]	[DISK PRINT]	[STOR NOSTOR]	
[REP NOREP]	[SEQ8 NOSEQ8]	[INC NOINC]	[CTL NOCTL]	[OUTMODE <i>fm</i>]							
[STK NOSTK]	[TERM NOTERM]	[DISK PRINT]	[STOR NOSTOR]								

Operands

fn1 ft1 fm1

is the file identifier of the source input file. The file must consist of fixed-format records with sequence fields in the last eight columns (or the last five columns). If you do not specify file type or file mode, a default of ASSEMBLE A1 is assumed.

fn2 ft2 fm2

is the file identifier of the update file. If you specify the NOCTL option, the file must be a control file that lists the update files you wish to apply. The default file identifier is *fn1* CNTRL A1. This file can also contain requisite information (PREREQ, CO-REQ, and IF-REQ comments) for applying the updates. The default file identifier is *fn1* UPDATE A1. If the CTL option is specified, this file must be a control file that lists the update files to be applied; the default file identifier is *fn1* CNTRL A1.

Options

REP

creates an output source file with the same file name as the input source file. If the output file is placed on the same disk or directory as the input file, the input file is erased.

NOREP

retains the old file in its original form, and assigns a different file name to the new file, consisting of a dollar sign (\$) plus the first seven characters of the input file name (*fn1*).

SEQ8

specifies that the entire sequence field (the last eight columns of the file) contains an eight-digit sequence number on every record of source input.

NOSEQ8

specifies that the last eight columns of the file contain a three-character label field, followed by a five-digit sequence number.

Note: The CMS editor, by default, sequences source files with five-digit sequence numbers.

INC

increments sequence numbers in the last eight columns in each record inserted into the updated output file, according to specifications in UPDATE control statements.

NOINC

puts asterisks (*****) in the sequence number field of each updated record inserted from the update file.

CTL

specifies that *fn2 ft2 fm2* describes an update control file for applying multiple update files to the source input file.

Note: The CTL option implies the INC option.

NOCTL

specifies that a single update file is to be applied to the source input file.

OUTMODE *fm*

specifies that the files created by the UPDATE command will be written onto the disk or directory accessed as *fm*. You must access the disk or directory as a *read/write*, otherwise the UPDATE command stops processing. If you do not specify a file mode number when you specify *fm*, the file mode number defaults to 1. If you do not specify OUTMODE, then the files are put onto the disk or directory as outlined in a following section, File Mode of Output Files.

STK

stacks information from the control file in the CMS console stack. STK is valid only if you specify the CTL option, and is valid only when you issue the UPDATE command from an EXEC procedure.

NOSTK

does not stack control file information in the console stack.

TERM

displays warning messages at the terminal whenever UPDATE finds a sequence or update control card error. (Such warning messages appear in the update log, whether UPDATE displays them at the terminal or not.)

NOTERM

suppresses the display of warning messages at the terminal. However, error messages that stop the entire update procedure are displayed at the terminal.

DISK

places the update log file on a disk or directory. This file has a file identifier *fn* UPDLOG, where *fn* is the file name of the file being updated.

PRINT

prints the update log file directly on the virtual printer.

STOR

specifies that the source input file is to be read into storage and the updates performed in storage prior to placing the updated source file on a disk or directory. This option is meaningful only for use with the CTL option, since the benefit of increased processing speed is realized when processing multiple updates. STOR is the default when you specify CTL.

NOSTOR

specifies that no updating is to take place in storage. NOSTOR is the default when you apply single updates (i.e., you do not specify CTL on the command line).

Update Control Statements: The UPDATE control statements let you insert, delete, and replace source records, as well as resequence the output file.

All references to the sequence field of an input record refer to the numeric data in the last eight columns of the source record, or the last five columns if NOSEQ8 is specified. Leading zeros in sequence fields are not required. If no sequence numbers exist in an input file, a preliminary UPDATE with only the './ S' control statement can be used to establish file sequencing.

UPDATE checks the sequence numbers while applying updates. An error condition results if any sequence errors occur in the update control statements, and warnings are issued if an error is detected in the sequencing of the input file. UPDATE skips any source input records with a sequence field of eight blanks without any indication of a sequence error. UPDATE replaces or deletes such records only if they occur within a range of records that are being replaced or deleted entirely, and if that range has limits with valid sequence numbers. There is no means provided for specifying a sequence field of blanks on an UPDATE control statement.

Control Statement Formats: All UPDATE control statements are identified by the characters './' in columns 1 and 2 of the record, followed by one or more blanks and additional, blank-delimited fields. Control statement data must not extend beyond column 50.

SEQUENCE Control Statement: Numbers or rennumbers the records in a file. Sequence numbers are written in the last eight columns (if you specify SEQ8), or in the last five columns with the label placed in the three preceding columns (if you do not specify NOSEQ8).

The format of the SEQUENCE control statement is:

```
./ S [seqstrt [seqincr [label]]]
```

where:**seqstrt**

is a one- to eight-digit numeric field specifying the first decimal sequence number to be used. The default value is 1000 if SEQ8 is specified and 10 if NOSEQ8 is specified.

seqincr

is a one- to eight-digit numeric field specifying the decimal increment for resequencing the output file. The default is the "seqstr" value.

label

is a three-character field to be duplicated in the first three of the last eight columns of each source record if NOSEQ8 is specified. The default value is the first three characters of the input file name (*fn1*).

If you use the SEQUENCE statement, it must be the first statement in the update file. If any valid control statement precedes it, the resequence operation is suppressed.

When the sequence control statement is the first statement processed, the sequence numbers in the source file are checked and warning message DMSUPD210W is issued for any errors. If the sequence control statement is processed after updates have been applied, no warning messages will be issued.

Each source record is resequenced in the last eight columns as it is written onto the output file, including unchanged records from the source file and records inserted from the update file.

INSERT Control Statement: Precedes new records that you may want to add to a source file.

The format of the INSERT control statement is:

```
./ I seqno [$ [seqstr [seqincr]]]
```

*where:**seqno*

is the sequence number of the record in the source input file where you want to add new records.

\$

is an optional delimiter indicating to sequence the inserted records by increments.

seqstr

is a one- to eight-digit numeric field specifying the first decimal increment for sequencing the inserted records.

seqincr

is a one- to eight-digit numeric field specifying the decimal increment for sequencing the inserted records.

The INSERT statement tells UPDATE where to add the new records. For example, the lines:

```
./ I 1600
TEST2 TM   HOLIDAY,X'02'   HOLIDAY
          BNO  VACATION     NOPE...VACATION
```

insert two lines of code, following the statement numbered 1600, into the output file. The inserted lines are flagged with asterisks in the last eight columns (if you specify NOCINC). If you specify either the INC or CTL option, UPDATE inserts the

UPDATE

records unchanged in the output file, or they are sequenced according to the *seqstrt* fields, if you specify the dollar sign (\$) key.

The default sequence increment, if you include the dollar sign, is determined by using one tenth of the least significant, nonzero digit in the *seqno* field, with a maximum of 100. The default *seqstrt* is computed as *seqno* plus the default *seqincr*. For example, the control statement:

```
./ I 2600 $ 2610
```

causes the inserted records to be sequenced XXX02610, XXX02620, and so forth (NOSEQ8 is assumed here). For the control statement:

```
./ I 240000 $
```

the defaulted *seqincr* is the maximum, 100, and the starting sequence number is 240100. UPDATE assumes SEQ8, so it sequences the inserted records, 00240100, 00240200, and so forth.

If you specify either INC or CTL, not the dollar sign, whatever sequence number appears on the inserted records in the update file is included in the output file.

DELETE Control Statement: Deletes one or more records from the source file.

The format of the DELETE control statement is:

```
./ D seqno1 [seqno2] [$]
```

where:

seqno1

is the sequence number identifying the first or only record to be deleted.

seqno2

is the sequence number of the last record to be deleted.

\$

is an optional delimiter indicating the end of the control fields.

All records of the input file, beginning at *seqno1*, up to and including the *seqno2* record, are deleted from the output file. If you do not specify *seqno2*, then UPDATE deletes only a single record.

REPLACE Control Statement: Replaces one or more input records with updated records from the update file. It precedes any new records you may want to add. It is a combination of the DELETE and INSERT statements.

The format of the REPLACE control statement is:

```
./ R seqno1 [seqno2] [$ [seqstrt [seqincr]]]
```

where:

seqno1

is the sequence number of the first input record to be replaced.

seqno2

is the sequence number of the last record to be replaced.

\$

is an optional delimiter key indicating that the substituted records are to be sequenced incrementally.

seqstrt

is a one- to eight-digit numeric field specifying the first decimal number to be used for sequencing the substituted records.

seqincr

is a one- to eight-digit numeric field specifying the decimal increment for sequencing the substituted records.

For example, the lines:

```
./ R 38000 38500
PLIST DS OD
      DC CL8X'TYPE'
      DC CL8X' '
      DC CL8X'FILE'
      DC CL8X'A1'
      DC 8XX'FF'
```

replace the existing statements numbered 38000 through 38500 with the new lines of code. As with the INSERT statement, UPDATE does not automatically resequence new lines. In addition, the dollar sign (\$), *seqstrt*, and *seqincr* processing is identical to that for the INSERT statement.

COMMENT Statement: Lets you place comments in the update log file. Note that the COMMENT statement is treated as a control statement when it appears within a sequence of records to be applied in an update.

The format of the COMMENT statement is:

```
./ * [comment]
```

where:

*

indicates that this is a comment statement. PREREQ, CO-REQ, and IF-REQ comments identify requisite updates. PREREQs and CO-REQs describe dependencies in the same product. IF-REQs describe dependencies in other products. All comments are copied into the update log file. PREREQ, CO-REQ, and IF-REQ comments are also copied into the *fn* UPDATES file.

Summary of Files Used by the UPDATE Command: The following discussion shows input and output files used by the UPDATE command for a:

- Single update
- Multiple updates
- Multiple updates with an auxiliary control file

File Mode of Output Files: The following is a list of steps that determines the disk or directory selection for placing the output files (the search stops as soon as one of these steps is successful):

1. If you specify the OUTMODE option, then UPDATE places the output files on the disk or directory specified, if it accessed as read/write. If the disk or directory you specify is accessed as a read/only extension, then the following message is displayed:

```
DMSUPD037E   Filemode mode [(vdev)] is accessed as read/only
```

2. If the disk or directory on which the original source file resides is read/write, then UPDATE places the output files on that disk or directory.
3. If the disk or directory on which the original source file resides is a read-only extension of a read/write disk or directory, then UPDATE places the output files on that particular read/write disk or directory.
4. If neither of the last two steps is successful, then UPDATE places the output files on the disk or directory accessed as A.

Applying a Single Update: Let's say that you have created an update file, *fn* UPDATE, and you want to apply it to the source file *fn* ASSEMBLE. You can do this by issuing the following command:

```
update fn
```

UPDATE makes the changes to the source file that you indicate in the update file and creates an updated version of the source file, but with a different file name. By default, the updated version of the file is called *\$fn* ASSEMBLE. UPDATE also creates a file, *fn* UPDLOG, which is a record of the updates applied. For the above example, if you do not want this update log file written on a disk or directory, specify the PRINT option.

Note: You can override the default file types and file modes of the output files on the command line. For example,

```
update testprog cobol b fix cobol b (rep
```

results in updating the source file TESTPROG COBOL B, with control statements contained in the update file FIX COBOL B. The output file replaces the existing TESTPROG COBOL B.

Using UPDATE with a Control File: If you have more than one update file that you want to apply to the same source file, you can apply them using a series of single updates, or you can use the UPDATE command specifying a control file. A control file lists the file type of the update files that you want to apply to a source file. The control itself does not contain the actual UPDATE control statements.

Let's say you have two update files, *fn* UPDTABC and *fn* UDPTXYZ, and that they contain UPDATE control statements and new source records. These two update files must have file names that are the same as the source input file. The first four characters of the file type must be UPDT. The UPDATE command searches all accessed disks and directories to locate the update files. You can use one UPDATE command to apply these updates to one file at the same time. You do this by specifying a control file.

A control file (*fn* CNTRL) lists the file types of the files that contain UPDATE control statements. As an example, let's use the following sample control file:

```
*THIS IS A SAMPLE CNTRL FILE
TEXT MACS CMSLIB
TWO UPDTABC
ONE UDPTXYZ
```

- **TEXT**, **TWO**, and **ONE** are the characters in the first column of each line of the control file called update level identifiers. Each update level identifier can be from one to five characters long. VM/SP updating procedures such as the VMFSAM EXEC use these identifiers to locate and identify text decks produced by multi-level updates.
- **MACS** must be the first non-commentary record in the control file. It contains an update level identifier (TEXT) and, optionally, lists up to 29 macro library (MACLIB) file names (subject to the record length of the line). If the list does not fit on one MACS record, additional MACS records can be included. Each additional MACS record must have the same format as the first one. However, the update level identifier on additional MACS records is ignored. All MACS records must be contiguous, and the total length of all macro library names plus separating blanks must be less than or equal to 253 characters. Any MACLIB names that extend beyond this limit will be truncated. The length of the MACS record may not exceed 80, the record length of the control file.

UPDATE uses the information provided in the MACS card and the update level identifier only when you specify the STK option. This information is, however, required in the CNTRL file.

- **UPDTABC**, and **UPDTXYZ** are file types of the update files. The UPDATE command applies these updates to the source file beginning with the last record in the control file. Thus, the updates in *fn* UPDTXYZ are applied before the updates in *fn* UPDTABC.

These files can also contain PREREQ, CO-REQ, and IF-REQ comments that specify dependencies for applying the updates. The update files must have file names that are the same as the source input file.

So, in our example, to update *fn* ASSEMBLE with our sample control file, issue the following:

```
update fn (ctl)
```

UPDATE looks at the control file and begins applying the updates listed starting with *fn* UPDTXYZ (the bottom entry).

When you create update files that have file types beginning with UPDT, you may omit these characters when you list the updates in the control file; thus, the CNTRL file may be written:

```
TEXT MACS CMSLIB
TWO ABC
ONE XYZ
```

file, and includes all requisite information from the update files. The CONTROL FILE (*fn* CNTRL) may not contain UPDATE control statements. It may only list the file types of the files that contain UPDATE

Using UPDATE with an Auxiliary Control File: There may be times when you have two groups of programmers working on different sets of changes for the same source file. Each group may create several update files and have a unique control file. When you combine these changes, you could create one control file or you can use *auxiliary control* files. An auxiliary control file is a list of file types of the update files you want to apply to a source file.

UPDATE

Let's take an example where you want to make an update using an auxiliary control file. You have *fn* ASSEMBLE as the source file, *fn* UPDTABC and *fn* UPDTXYZ, as the update files, and a control file that looks like the following:

```
TEXT MACS CMSLIB
TWO UPDTABC
ONE UPDTXYZ
TEXT AUXLIST
```

- *AUX* in the file type AUXLIST indicates that this is an auxiliary control file. This is another type of file listing the file types of update files you want to apply to a source file. In this example, the auxiliary control file lists the update files FIX01 and FIX02.

When you issue:

```
update fn (ctl
```

UPDATE looks in the control file at the bottom entry, TEXT AUXLIST. Since this is an auxiliary control file, UPDATE looks in it and applies the updates listed in it (starting with the bottom entry) before applying the other updated files listed in the control file *fn* CNTRL.

Note that the file name of an auxiliary control file must be the same as the source input file. The file type must begin with the characters AUX and the remaining characters (a maximum of five) can be anything.

You may also specify an auxiliary file as:

```
xxxxx AUX
```

in the control file. For example, the record:

```
FIX TEST AUX
```

identifies the auxiliary file *fn* AUXTEST.

Note that if you give an auxiliary control file the file type AUXPTF or an update level identifier of AUX, the UPDATE command assumes that it is a simple update file and does not treat it as an auxiliary file.

Additional Control File Records: In addition to the MACS record, the file types of update (UPDT) files, and the file types of auxiliary control (AUX) files, a control file may also contain:

- Comments. These records begin with an asterisk (*) in column 1. Comments are also valid in AUX files.
- PTF records. If the characters PTF appear in the update level identifier field, the UPDATE command expects the second field to contain the file type of an update file. The file type may be anything; the file name must be the same as the source input file.
- Update level identifiers not associated with update files.

The following example of a control file shows all the valid types of records:

```
* Example of a control file
ABC MACS MYLIB
TEXT
004 UPDTABC
003 XYZ
002 AUXLIST1
001 LIST2 AUX
PTF TESTFIX
```

Preferred AUX File: By using preferred auxiliary control files, you can use one control file for multiple versions of a file or multiple releases of a product. (There may be more than one version of the same update if there is more than one version of the source file. For example, you need one version for the source file that has a system extension licensed program installed, and you need another version for the source file that does not have a licensed program installed.)

Remember, to specify *auxiliary* control files, specify their file type in a control file.

To use *preferred auxiliary* control files, you must specify more than one file type per update level identifier. The first file type indicates a file that UPDATE will use if *none* of the additional file types exist for any disks or directories you have accessed. If any of the files indicate that the additional file types do exist, then UPDATE ignores that entire entry and precedes to the next entry in the control file. The files that indicate additional file types are preferred because UPDATE does not use the file that the first file type indicates. For example, assume that you want to update the file SAMPLE ASSEMBLE using the updates in SAMPLE AUXTEST. To update SAMPLE ASSEMBLE, use the following control file (MYPROG CNTRL):

```
TEXT MACS MYMACS CMSLIB OSMACRO
MY2 AUXTEST
MY1 AUXMINE AUXTEST
```

and the command:

```
UPDATE SAMPLE ASSEMBLE * MYMODS CNTRL (CTL
```

UPDATE looks at the bottom entry in the control file first. It searches all accessed disk and directories for the auxiliary control file SAMPLE AUXTEST. Since that file exists, UPDATE does not use the auxiliary control file SAMPLE AUXTEST. Instead, UPDATE ignores this entry and continues to the next entry in the control file. The next entry only specifies one file type (AUXTEST). This is the *preferred* AUX file you want to use, so UPDATE applies the updates listed in SAMPLE AUXTEST. It is assumed that AUXTEST and AUXMINE list similar but mutually exclusive updates.

The search for a *preferred* auxfile will continue until one is found or until the token is an invalid file type; that is, less than four or more than eight characters. This token and the remainder of the line are considered a comment.

The *fn* FIX01 and *fn* FIX02 are update files containing UPDATE control statements and new source records to be incorporated into the input file. These files can also contain PREREQ, CO-REQ, and IF-REQ comments that specify dependencies for applying the updates.

UPDATE

The STK Option: The STK (stack) option is valid only with the CTL option and is meaningful only when the UPDATE command is issued within an EXEC procedure.

When the STK option is specified, UPDATE stacks the following data lines in the console stack:

first line: * update level identifier
second line: * library list from MACS record

The update level identifier is the identifier of the most recent update file that was found and applied. For example, if a control file contains

```
TEXT MACS CMSLIB OSMACRO TESTMAC
OFA UPDFOFA
PFA UPDFOFA
```

and the UPDATE command appears in an exec as follows:

```
UPDATE SAMPLE (CTL STK
&READ VARS &STAR &TEXT
&READ VARS &STAR &LIB1 &LIB2 &LIB3 &LIB4
```

then the variable symbols set by the &READ VARS statements have the following values if the file SAMPLE UPDFOFA is found and applied to the file SAMPLE ASSEMBLE:

<i>Symbol</i>	<i>Value</i>
&STAR	*
&TEXT	OFA
&LIB1	CMSLIB
&LIB2	OSMACRO
&LIB3	TESTMAC
&LIB4	null

The library list may be useful to establish macro libraries in a subsequent GLOBAL command within the EXEC procedure. If no update files are found, UPDATE stacks the update level identifier on the MACS record.

Responses

FILE *fn ft fm* REC #*n* = update control statement

This message is displayed when the TERM option is specified and an error is detected in an update file. It identifies the file and record number where the error is found.

DMSUPD177I Warning messages issued (severity = *nn*); REP option ignored]

Warning messages were issued during the updating process. The severity shown in the error message in the “nn” field is the highest of the return codes associated with the warning messages that were generated during the updating process.

The warning return codes have the following meanings:

RC = 4 - Sequence errors were detected in the original source file being updated.

RC = 8 - Sequence errors, which did not previously exist in the source file being updated, were introduced in the output file during the updating process.

RC = 12 - Any other warning error detected during the updating process. Such errors include invalid update file control statements and missing update or PTF files.

The severity value is passed back as the return code from the UPDATE command. In addition, if the REP option is specified in the command line, then it is ignored, and the updated source file has the file ID \$*fn1 ft1*, as if the REP option was not specified.

```
DMSUPD178I Updating fn ft fm
           Applying fn ft fm
           Applying fn ft fm
           .
           .
           .
```

The specified update file is being applied to the source file. This message appears only if the CTL option is specified in the command line. The updating process continues.

DMSUPD304I Update processing will be done using disk

An insufficient amount of virtual storage was available to perform the updating in virtual storage, so a CMS disk or SFS directory must be used. This message is displayed only if NOSTOR was specified in the UPDATE command line.

DMSUPD180W Missing PTF file *fn ft fm* RC = 12

In the event that the user receives this message during the update process, the message MISSING PTF FILE *fn ft fm* will appear in the update log associated with the program being updated.

Messages and Return Codes

DMSUPD001E	No filename specified [RC = 24]
DMSUPD002E	File [<i>fn [ft [fm]]</i>] not found [RC = 28]
DMSUPD003E	Invalid option: <i>option</i> [RC = 24]
DMSUPD007E	File <i>fn ft fm</i> is not fixed, 80-character records [RC = 32]
DMSUPD007E	File <i>fn</i> does not have the same format and record length as <i>fn</i> [RC = 32]
DMSUPD007E	File <i>fn ft fm</i> is not fixed record format [RC = 32]
DMSUPD007E	File <i>fn ft fm</i> does not have a logical record length greater than or equal to 80 and less than or equal to 255 [RC = 32]
DMSUPD010W	Premature EOF on file <i>fn ft fm</i> --sequence number <i>seqno</i> not found [RC = 12]
DMSUPD024E	File <i>fn ft fm</i> already exists [RC = 28]
DMSUPD048E	Invalid mode <i>mode</i> [RC = 24]
DMSUPD065E	<i>option</i> option specified twice [RC = 24]
DMSUPD066E	<i>option1</i> and <i>option2</i> are conflicting options [RC = 24]
DMSUPD069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSUPD070E	Invalid parameter <i>parameter</i> [RC = 24]
DMSUPD104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC = 100]
DMSUPD105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSUPD174W	Sequence error introduced in output file: <i>seqno1</i> to <i>seqno2</i> <i>seqno1</i> to <i>seqno2</i> [RC = 8]
DMSUPD176W	Sequencing overflow following sequence number <i>seqno</i> [RC = 8]
DMSUPD179E	Missing or invalid MACS card in control file <i>fn ft fm</i> [RC = 32]
DMSUPD181E	No update files were found [RC = 40]
DMSUPD182W	Sequence increment is zero [RC = 8]
DMSUPD183E	Invalid {CONTROL AUX} file control card [RC = 32]
DMSUPD184W	./S not first card in update file--ignored [RC = 12]
DMSUPD185W	Invalid character in sequence field <i>seqno</i> [RC = 12]
DMSUPD186W	Sequence number <i>seqno</i> not found [RC = 12]
DMSUPD187E	Option STK invalid without CTL [RC = 24]

UPDATE

DMSUPD207W Invalid update file control card [RC=12]
DMSUPD210W Input file sequence error: *seqno1* to *seqno2* [RC=4]
DMSUPD299E Insufficient storage to complete update [RC=41]
DMSUPD300E Insufficient storage to begin update [RC=41]
DMSUPD361E Disk *mode* is not a CMS disk [RC=36]
DMSUPD1259E File pool *filepoolid* has run out of physical space in the storage group [RC=40]
DMSUPD1262E Error *nn* opening file *fn ft fm* [RC=*nn*]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

VALIDATE

Use the VALIDATE command to verify the syntax of a file identifier (file name, file type, file mode). If you specify the file mode, VALIDATE verifies whether or not the disk or directory is accessed.

Format

VALIDATE	$\left\{ \begin{array}{c} fn \\ * \end{array} \right\}$	$\left\{ \begin{array}{c} ft \\ * \end{array} \right\}$	$\left[\begin{array}{c} fm \\ - \\ * \end{array} \right]$
-----------------	---	---	--

Operands

fn

is the file name whose syntax is to be verified. If *fn* is specified as *, it is ignored.

ft

is the file type whose syntax is to be verified. If *ft* is specified as *, it is ignored.

fm

is the file mode whose syntax is to be verified. If *fm* is specified, the disk or directory will be checked for access. If *fm* is omitted, or specified as *, no disks or directories are checked for access.

Usage Notes

1. The file name and file type can each be one to eight characters. The valid characters are A-Z, a-z, 0-9, \$, #, @, +, - (hyphen), : (colon), and _ (underscore).
2. When you code an asterisk in the *fn* or *ft* fields, only the specified field will be verified. For example, the command:


```
validate * file e
```

 verifies the syntax of the file type FILE and determines if a disk or directory is accessed as E.
3. You can invoke the VALIDATE command from the terminal, from an exec, or as a function from a program. If VALIDATE is invoked from an exec or as a function that has the message output suppressed, the messages are not issued.
4. When writing execs or assembler programs, you can use VALIDATE * * *fm* to determine if a disk or directory is accessed. For example,


```
validate * * e
```

 tells you if the disk or directory at E is accessed, regardless if any files exist on it. If it is not accessed, an error message is issued.
5. To verify the syntax of a file identifier and the existence of the file on an accessed disk or directory, use the STATE/STATEW (or ESTATE/ESTATEW) command.

VALIDATE

Responses

The CMS ready message indicates that the specified file identifier is valid and the file mode is an accessed disk or directory or was specified as *.

Messages and Return Codes

DMSSTT048E Invalid mode *mode* [RC = 24]
DMSSTT054E Incomplete fileid specified [RC = 24]
DMSSTT062E Invalid character *char* in fileid *fn ft* [RC = 20]
DMSSTT069E Filemode *mode* not accessed [RC = 36]
DMSSTT070E Invalid parameter *parameter* [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command <i>syntax</i>	811

XEDIT

Use the XEDIT command to invoke the VM/SP System Product Editor to create, modify, and manipulate CMS files on minidisks or in Shared File System (SFS) directories. Once the VM/SP System Product Editor has been invoked, you may execute XEDIT subcommands and use the System Product Interpreter or EXEC 2 macro facility.

You can return control to the CMS environment by issuing one of the XEDIT subcommands: FILE, FFILE, QUIT, or QQUIT.

For complete details on XEDIT subcommands and macros, see the publication *VM/SP System Product Editor Command and Macro Reference*.

Format

Xedit	<p>[<i>fn</i> [<i>ft</i> [<i>fm</i>]]] [(options...)]</p> <p>Options:</p> <p>[WINdow <i>wname</i>] [Width <i>nn</i>] [NOSCrEen] [PROfile <i>macroname</i>] [NOPROF<i>il</i>] [NOCLear] [NOMsg] [MEMber <i>membername</i>] [LOCK] [NOLOCK]</p> <p>Options Valid Only in Update Mode:</p> <p>[Uppdate] [Seq8] [Ctl <i>fnl</i>] [NOUpdate] [NOSeq8] [NOctl] [Merge] [UNtil <i>filetype</i>] [Incr <i>nn</i>] [SIDcode <i>string</i>]</p>
--------------	---

Operands

fn ft

are the file name and file type of the file to be edited. If they are not specified here, they must be provided in the LOAD subcommand as part of the profile.

fm

is the file mode of the file to be edited, indicating the minidisk or directory where the file resides. The editor determines the file mode of the edited file as follows:

- Editing existing files

When the file mode is specified, that minidisk or directory and its extensions are searched. If the file mode is not specified or is specified as an asterisk (*), all accessed minidisks and directories are searched for the specified file.

- Creating new files

If the file mode is not specified, the editor assumes a file mode of A1.

Options**WINDow** *wname*

is the name of the window and virtual screen that XEDIT uses to display the file being edited. By default, XEDIT uses the window and virtual screen named "XEDIT." For more information, refer to Appendix G of the *VM/SP System Product Editor Command and Macro Reference*.

Width *nn*

defines the amount of virtual storage used to contain one line of the file. If the value specified is too small, certain file lines may be truncated.

If not specified here, WIDTH may be defined in the LOAD subcommand, as a part of the profile. If WIDTH is not specified in either the XEDIT command or the LOAD subcommand, the default is the larger of the following:

- The logical record length (LRECL) of the file
- The default logical record length associated with the file type.

NOSCrEen

forces a 3270 display terminal into line (typewriter) mode.

PROFile *macroname*

If the specified macro exists on one of the accessed minidisks or directories, the editor executes it as the first subcommand.

If the specified macro is not found on an accessed minidisk or directory, an error message is displayed.

If this option is not specified but a macro with a macro name of PROFILE exists, the editor executes it.

In all cases, the profile macro must have a file type of XEDIT.

NOPROFIl

forces the editor not to execute the default PROFILE macro.

NOCLeAr

specifies that the screen is not cleared when the editor gets control. Instead, the screen is placed in a MORE... (waiting) status. Any messages remain on the screen until the CLEAR key is pressed. This option is useful when the XEDIT command is issued from a macro that displays messages.

NOMsg

enters a file with a default of SET MSGMODE OFF.

MEMber *membername*

specifies the name of a member to be edited specified in 'fn ft fm' macro library. If MEMBER is specified, ft must be MACLIB. If the member does not exist in that library, a new file is created with a file ID of '*membername* MEMBER *fm*'.

LOCK

causes the editor to lock the file to prevent other users from modifying the file while you are editing it. Only existing files in SFS directories can be locked; the LOCK option is ignored for files on minidisks. You must have write authority to the file to lock it; if you have only read authority, a warning is displayed and the editing session continues without locking the file. LOCK is the default.

The type of lock XEDIT uses is an update session lock. The file is locked only for the duration of your editing session. Other users can read the file while it is locked, but only you can write to it.

NOLOCK

indicates that you do not want the editor to lock the file. This option can be used to edit a file that another user has locked SHARE or UPDATE. If you specify NOLOCK, other users may change the file while you are editing it. The NOLOCK option is ignored for files on minidisks.

You should only use this option if you are not going to make any changes to the file, or if you are going to save your changes under a different file identifier. Otherwise, any changes that you make will not include modifications made to the permanent copy of the file by other users during your editing session.

Options - (Use in Update Mode Only)

The following options are meaningful only if the VM/SP System Product Editor is to be used in update mode:

Update

The editor searches all accessed minidisks and directories for a file with a file name of *fn* and a file type of UPDATE. If the file exists, the editor applies the update statements before displaying the file to be edited. Each new modification made by the user is added to the existing UPDATE file. The original source file is *not* modified.

If the file does not exist, the editor creates a new UPDATE file to contain modifications made by the user.

NOUpdate

specifies that the editor is to apply no update statements (even if UPDATE is specified in the LOAD subcommand in the profile).

Seq8

specifies that the entire sequence field (the last eight columns of each file line) contains an eight-digit sequence number. The SEQ8 option automatically forces the UPDATE option. SEQ8 is the default value.

NOSeq8

specifies that the last eight columns of the file line contain a three-character label field, followed by a five-digit sequence number.

The NOSEQ8 option forces the UPDATE option.

Ctl *fn1*

specifies that “*fn1* CNTRL” is an update control file that controls the application of multiple update files to the file to be edited. (See the CMS UPDATE command description for more information.)

This option automatically forces the UPDATE and SEQ8 options.

NOCtl

specifies that the editor is not to use the control (CTL) file (even if it is specified in the LOAD subcommand in the profile).

Merge

specifies that all updates, and all changes made while editing are recorded into a file whose name is defined by the latest update level (in other words, the most recently applied UPDATE file in a control file). This option forces the UPDATE option.

UNtil *filetype*

specifies the file type of the last update to be applied to the file. Changes are applied to the file being edited from all file types in the control file, up to and including the *filetype* specified with the UNTIL option.

File types of update files listed in the control file or of update files listed in an auxiliary control file can be specified with the UNTIL option. AUX file types (AUXxxxxx) cannot be specified with the UNTIL option.

The UNTIL option forces the UPDATE option.

Incr *m*

means the minimum number of lines to automatically be inserted. When inserting new lines in an update file, the editor automatically computes the serialization; the INCR option forces a minimum increment between two adjacent lines. If not specified, the minimum increment is one (1). This option forces the UPDATE option.

SIDcode *string*

specifies a string that the editor inserts in every line of an update file whether the update file is an existing file or if it is being created. The editor inserts the specified string in the first eight columns of the last 17 columns of the file (lrecl-16 to lrecl-9). For example, if your file is fixed, 80-character, the editor inserts the string in columns 64-71. If the string is less than eight characters, it is padded on the right with blanks. Any data in the eight columns is overlaid. This option forces the UPDATE option.

Usage Notes

1. In order to use XEDIT on files in SFS directories, the directories must be accessed and you must have read or write authority to the file.
2. When the LOCK option is in effect, it is possible for the lock to be removed during your editing session if one of the following abnormal error occurs:
 - file system server failure
 - network or APPC/VM failure on the last communication link with the file pool.

Note: If an abend occurs, the update session lock obtained by XEDIT will not be deleted.
3. For the CTL, INCR, MEMBER, PROFILE, SIDCODE, UNTIL, and WIDTH options, the operand must be specified; otherwise, the next option will be interpreted as the operand. For example, in the "PROFILE macroname" option, "macroname" must be specified; if it is not, the next option will be interpreted as the operand macro name.
4. Once the XEDIT *command* has been executed, the XEDIT *subcommand* can be used to edit and display multiple files simultaneously. See the XEDIT subcommand description in the publication *VM/SP System Product Editor Command and Macro Reference*.
5. You can also call the editor recursively (using "CMS XEDIT..." for example). This ability is particularly useful when applications are developed using the editor and its macro facilities to interface with the user - for example, HELP.
6. The MEMBER option and the NOUPDATE option have no effect when preceded by an option that automatically forces update processing. Likewise, options that normally force update processing are ignored when preceded by the MEMBER option or the NOUPDATE option.
7. If FULLSCREEN is set to ON before XEDIT writes to the screen, XEDIT issues the command, SHOW WINDOW CMSOUT, followed by SHOW WINDOW XEDIT, or a SHOW for the particular window that is set up to display the file.

8. The editor is kept in virtual storage as part of the CMS nucleus shared segment; the CMS user area is unused. As a result, assuming a large enough virtual machine, any CMS or CP command may be issued directly from the editor environment itself (if a SET IMPCMSCP subcommand is in effect).
9. When the PROFILE macro is invoked by an XEDIT command, everything following the command name XEDIT is tokenized and then assigned to the argument string that is passed to the PROFILE macro. The editor does not examine any parameters that follow a closing right parenthesis on the XEDIT command.
10. When you issue an XEDIT command for a variable-format file, trailing blanks are removed when the file is filed (or saved).
11. Comment control records are deleted from an update file whenever an update file is applied to the original source file during an editing session, and a FILE or SAVE subcommand is issued.
12. Many languages have more characters than can be displayed using one-byte codes (KANJI, for example). A Double-Byte Character Set (DBCS) is used to represent those characters. The double-byte characters may appear in a sentence with characters from other languages that are displayed in one-byte codes. Files containing double-byte characters are handled differently than files that only contain one-byte characters. Special considerations for editing files that contain double-byte characters are described in the *System Product Editor Command and Macro Reference*.

Responses

When editing a file that resides in an SFS directory, if you have read authority to the file and you do not specify the NOLOCK option, you will receive the message:

```
DMSXIN1299W Warning: Not authorized to lock fn ft fm
```

The editing session continues; however, the file is not locked.

The following messages are displayed only if you are using the System Product Editor in update mode:

```
DMSXUP178I Updating fn ft fm
      Applying fn ft fm
      Applying fn ft fm
```

```
      .
      .
      .
```

```
DMSXUP180W Missing PTF file fn ft fm
```

Messages and Return Codes

DMSXBG109S	Virtual storage capacity exceeded [RC = 104]
DMSXFI1138E	File sharing conflict for file <i>fn ft fm</i> [RC = 70]
DMSXFI1214W	File <i>fn ft fm</i> already locked SHARE
DMSXFI1215E	File <i>fn ft fm</i> is locked {SHARE UPDATE EXCLUSIVE} by another user. [RC = 70]
DMSXFI1262S	Error <i>nn</i> opening file <i>fn ft fm</i> [RC = 31, 55, 70, 76, 99, or 100]
DMSXFI1300E	Error <i>nn</i> {locking unlocking} file <i>fn ft {fm dirname}</i> [RC = 55, 70, 76, 99, or 100]
DMSXIN002E	File <i>fn ft fm</i> not found [RC = 28]
DMSXIN003E	Invalid option: <i>option</i> [RC = 24]
DMSXIN024E	File XEDTEMP CMSUT1 A1 already exists [RC = 28]
DMSXIN029E	Invalid parameter <i>parameter</i> in the option <i>option</i> field [RC = 24]

DMSXIN054E Incomplete fileid specified [RC=24]
 DMSXIN065E *option* option specified twice [RC=24]
 DMSXIN066E *option1* and *option2* are conflicting options [RC=24]
 DMSXIN070E Invalid parameter *parameter* [RC=24]
 DMSXIN104S Error *nn* reading file *fn ft fm* from disk [RC=31, 55, or 100]
 DMSXIN132S File *fn ft fm* too large [RC=88]
 DMSXIN500E Unable to unpack file *fn ft fm* [RC=88]
 DMSXIN508E LOAD must be the first subcommand in the profile [RC=3]
 DMSXIN554E Not enough virtual storage available [RC=104]
 DMSXIN571I Creating new file:
 DMSXIN622E Insufficient free storage for
 {MSGLINE|PFKEY/PAKEY|synonyms}
 DMSXIN928E Command is not valid for virtual screen CMS [RC=12]
 DMSXIN1299W Warning: Not authorized to lock file *fn ft fm*
 DMSXSU048E Invalid mode *mode* [RC=24]
 DMSXSU062E Invalid character in fileid *fn ft fm* [RC=20]
 DMSXSU069E Filemode *mode* not accessed [RC=36]
 DMSXSU229E Unsupported OS dataset, error *nn* [RC=80, 81, 82, or 83]
 DMSXWS915E Maximum number of windows already defined [RC=13]
 DMSXWS927E The virtual screen must contain at least 5 lines and 20 columns.
 [RC=24]

Messages with MEMBER option

DMSXMB007E File *fn ft fm* is not fixed, 80-character records [RC=32]
 DMSXMB033E File *fn ft fm* is not a library [RC=32]
 DMSXMB039E No entries in library *fn ft fm* [RC=32]
 DMSXMB167S Previous MACLIB function not finished [RC=88]
 DMSXMB622E Insufficient free storage for reading map [RC=104]

Messages with UPDATE options

DMSXUP007E File *fn ft fm* is not fixed, 80-character records [RC=32]
 DMSXUP007E File *fn ft fm* does not have a logical record length greater than or
 equal to 80 [RC=32]
 DMSXUP007E File *fn ft fm* does not have the same format and record length as
fn ft fm [RC=32]
 DMSXUP007E File *fn ft fm* is not fixed record format [RC=32]
 DMSXUP104S Error *nn* reading file *fn ft fm* from disk [RC=31, 32, or 55]
 DMSXUP174W Sequence error introduced in output file: *seqno1* to *seqno2*
 [RC=32]
 DMSXUP178I Applying *fn ft fm*
 DMSXUP179E Missing or invalid MACS card in control file *fn ft fm*
 DMSXUP180W Missing PTF file *fn ft fm*
 DMSXUP183E Invalid {CONTROL|AUX} file control card [RC=32]
 DMSXUP184W ./ S not first card in update file--ignored [RC=32]
 DMSXUP185W Non numeric character in sequence field *seqno* [RC=32]
 DMSXUP186W Sequence number [*seqno*] not found [RC=32]
 DMSXUP207W Invalid update file control card [RC=32]
 DMSXUP210W Input file sequence error: *seqno1* to *seqno2* [RC=32]
 DMSXUP570W Update *ft* specified in the UNTIL option field not found
 DMSXUP597E Unable to merge updates containing ./S cards [RC=32]
 DMSXUP1262S Error *nn* opening file *fn ft fm* [RC=31, 55, 70, 76, 99, or 100]

Return Codes

- 0 Normal
- 3 LOAD must be the first subcommand
- 6 Subcommand rejected in the profile due to LOAD error or QUIT subcommand has been issued in macro called from the last file in the ring
- 12 Command is not valid for virtual screen
- 13 Maximum number of windows already defined
- 20 Invalid character in file name or file type
- 24 Invalid parameters, or options
- 28 Source file not found (UPDATE MODE), or library not found (MEMBER option), or specified PROFILE macro does not exist, or file XEDTEMP CMSUT1 already exists
- 31 A rollback occurred
- 32 Error during updating process, or file is not a library, or library has no entries, or file is not fixed 80-character records, or updates do not match input file format
- 36 Corresponding minidisk or directory not accessed
- 55 APPC/VM communications error
- 70 File sharing conflict
- 76 Connection error
- 80 An I/O error occurred while an OS data set or DOS file was being read, or an OS or DOS disk was detached without being released
- 81 The file is an OS-read password-protected data set, or a DOS file with the input security indicator on
- 82 The OS data set or DOS file is not BPAM, BSAM, or QSAM
- 83 The OS data set or DOS file has more than 16 user labels or data extents
- 88 File is too large and does not fit into storage or previous MACLIB function was not finished or MACLIB limit exceeded
- 99 A required system resource is not available
- 100 Error reading the file into storage
- 104 Insufficient storage available to read library map

XMITMSG

Use the XMITMSG command (TRANSMIT MESSAGE) to retrieve a message from a CMS message repository file or your own message repository file. You supply a message identifier and substitution information, and XMITMSG gets the message that you requested. XMITMSG can be used in a REXX, EXEC 2, or CMS environment.

Format

XMITMSG	<p><i>msgnumber</i> [<i>sublist</i>] [(options... [])]</p> <p>Options:</p> <p>[FORmat <i>nn</i>] [LINE <i>nn</i>] [LETter <i>a</i>]</p> <p>[APPLID <i>applid</i>] [CALLER <i>name</i>] [VAR]</p> <p>[COMPress] [HEADer] [DISPlay]</p> <p>[NOCOMPress] [NOHEADer] [NODISPlay]</p> <p>[ERRMSG]</p> <p>[SYSLANG]</p>
----------------	---

Operands

msgnumber

is a one to four digit number used to locate its associated message text in the repository.

sublist

specifies the substitutions to be done on the message.

Any numeric substitution is assumed to be a dictionary substitution, and the substitution is retrieved from the repository. If the substitution is in either single or double quotes, then it is assumed to be a literal substitution. Literal substitutions must not contain blanks or parentheses. Any other substitution in the list is assumed to be a variable name, and the value of the substitution is retrieved from the exec. If the value cannot be retrieved, then the substitution is assumed to be null.

A maximum of 20 substitutions is allowed.

Options

FORmat *nn*

specifies a two-digit format number. It is used to identify different versions of the same message which have the same message number. The numbering of formats is from 01 to 99. The default is 01. A format of 00 is not allowed.

LINE *nn*

*

specifies a two-digit line number that identifies each line of a multi-line message. The numbering of lines is from 01 to 99. You may also specify an asterisk for the line number; this specifies that all lines for a certain message number and format are to be retrieved. The default line number is asterisk. A line number of 00 is not allowed. Each line may be up to 240 characters long.

LETter *a*

specifies the severity letter of the message. Message severity is provided already within the message repository module; this parameter is used only when you want to override the provided severity.

APPLID *applid*

specifies the name of the application from which the message is issued. The application ID is displayed in the message header.

CALLER *name*

overrides the default CALLER name that will go in the message header.

For execs, the default CALLER name is either:

1. the first three characters of the exec name, if these characters are different from the application id, or else
2. the next three characters of the exec name.

If you issue XMITMSG from the CMS command line, the default CALLER name is "???".

VAR

copies the message into variables and returns these variables to the exec. VAR is only valid when issued from an exec. The complete message is copied into the variable 'MESSAGE', with the first line in 'MESSAGE.1', the second in 'MESSAGE.2', etc..

The number of lines in the message is copied into 'MESSAGE.0'. If you do not specify NOHEADer, then the message header, i.e. DMSxxxxnnn or DMSxxxxnnn, is also part of the copied information. The length of the message is not returned to the exec.

If the message is to be displayed by the message facility and returned to the exec, then you must specify DISP. Otherwise, the default is NODISP when you specify VAR.

COMPress

removes multiple blanks in the message text, including those preceding and following a substitution field. This is the default for blank compression.

NOCOMPress

If NOCOMPRESS is specified on a message invocation with no substitutions, the message, as defined in the message repository, is not scanned. Therefore, no blanks will be replaced and no substitutions will be made. For example, if a message is defined in the repository with the substitution indicator &1, and the message is invoked with no substitutions and NOCOMPRESS, the &1 will appear in the displayed message.

To prevent the &1 substitution indicator from appearing in the message, a substitution must be specified on message invocation. This can be done by coding a NULL in the SUBLIST operand. If you specify a SUBLIST of " (null character) on the XMITMSG command, this causes the message text to be scanned and all substitution indicators are removed. For DBCS Languages

(created when GENMSG is issued with DBCS option), the message will be scanned in all cases.

HEADer

specifies that the message header is created and displayed with the message, or returned in MESSAGE.nn with the message text if VAR was specified.

NOHEADer

Specifies that the message header is not displayed on the terminal, or that it is not returned in MESSAGE.nn with the message text if VAR was specified. You may not specify this option with the ERRMSG option.

DISPlay

specifies that the message is displayed on the terminal, regardless of the CP EMSG setting. This is the default option unless you specify VAR.

NODISPlay

specifies that the message is not displayed on the terminal, regardless of the CP EMSG setting. This is the default option when you specify VAR.

ERRMSG

specifies that the message line is displayed according to the CP EMSG setting. If EMSG is set to:

- ON - the entire message is displayed, header plus text
- OFF - messages are not displayed
- TEXT - only the text portion is displayed
- CODE - only the header is displayed

The message header consists of the following:

xxxmmmmnnns or xxxmmmmnnns

where:

- xxx is the application id ("DMS" for CMS)
- mmm is the CALLER name
- nnn or nnnn is the message number
- s is the severity code

<i>Code</i>	<i>Message Type</i>
E	Error
I	Information
R	Response
S	Severe
T	Terminal
W	Warning

You cannot specify ERRMSG with the NOHEADER option.

SYSLANG

specifies that the system default language is to be used to issue the message, regardless of the language you are currently using. You may only specify this option when the application id is DMS.

Usage Notes

1. To learn how to create your own message repository refer to the *VM/SP Application Development Guide for CMS*.
2. You should have a copy of the message repository you want to access -- that way you can see the message numbers, formats, lines, and substitution positions.
3. You can use XMITMSG from CMS to display a repository message on your screen; this is useful when you want to verify the content of a repository.

Note: You cannot use the VAR option when invoking XMITMSG from CMS.

4. To issue messages from assembler programs, see the APPLMSG macro in the *VM/SP Application Development Reference for CMS*
5. If you are writing an editor macro and you want messages to be issued by the macro and not by XMITMSG, use the VAR and ERRMSG options. To prevent the message from being displayed in CMS rather than the editing environment, use the SET CMSTYPE HT command before the XMITMSG command, followed by SET CMSTYPE RT.

Examples

Assume the message repository for the application MYAPPL1 (*applid*=MYA) contains these messages:

```
08750101E Attempt to divide by &1 is invalid
08750201E Attempt to &2 by &1 is invalid
08760101E Error &1. rc = &3.
08770101E This is a multi-line message. NOCOMP must be specified
08770102E to keep the return codes lined up on the next line.
08770103E      RC 1 = &1.          RC 2 = &2.
| | |
| | |_____severity code
| | |_____line of message
| | |_____format of message
| | |_____number of message
```

and these dictionary items:

```
90250101 divide
90260101 reading from &2
90270101 tape
```

Following is an example of a REXX exec called DIVIDE that displays error messages when it attempts to divide by zero:

```
/* Example using XMITMSG in a REXX exec */
ARG DIVR
TEN = 10
IF DIVR = 0 THEN
DO
  ANS = TEN / DIVR
  EXIT
END
ELSE
DO
  ----- issue error message; see cases below -----
END
```


Case One:

This call accesses the repository to display MYA message 875, format 1, with '0' (the value in DIVR) being the substitution:

```
'XMITMSG 875 DIVR (DISP FOR 1 APPLID MYA COMP'
```

Here is what is displayed:

```
DMSDIV875E Attempt to divide by 0 is invalid
```

Note: The variable DIVR must be included *inside* the quotes.

Case Two:

This call uses a dictionary item 9025 as a second substitution in MYA message 875, format 2:

```
'XMITMSG 875 DIVR 9025 (DISP FOR 2 APPLID MYA COMP'
```

The same message is displayed as in Case One:

```
DMSDIV875E Attempt to divide by 0 is invalid
```

Case Three:

This call illustrates the use of the VAR option. Message 877 in MYA is accessed with two substitution values, 16 and RC2. Blanks are NOT compressed, so spacing is preserved.

```
RC2 = 8
```

```
'XMITMSG 877 "16" RC2 (APPLID MYA NOCOMP VAR'
```

The message is not yet displayed; instead, each line of the message is placed in a variable, 'MESSAGE.n'. The LINE parameter defaults to '*', meaning all lines of the message go into variables.

This code in the REXX program:

```
DO I = 1 TO MESSAGE.0 /* MESSAGE.0 = the number of message lines */
  SAY MESSAGE.I
END
```

displays the message:

```
DMSDIV877E This is a multi-line message. NOCOMP must be specified
           to keep the return codes lined up on the next line.
           RC 1 = 16.           RC 2 = 8.
```

Case Four:

This call again shows the use of the VAR option on MYA message 877, but only line 2 of the message is accessed:

```
'XMITMSG 877 (APPLID MYA NOCOMP VAR LINE 2'
```

This line in the REXX program:

```
SAY MESSAGE.1
```

then displays the following message line:

```
DMSDIV877E to keep the return codes lined up on the next line.
```

Messages and Return Codes

DMSMGX065E *option* option specified twice [RC=24]
 DMSMGX066E *option1* and *option2* are conflicting options [RC=24]
 DMSMGX080E Invalid *numtype* number [RC=24]
 DMSMGX109S Virtual storage capacity exceeded [RC=104]
 DMSMGX405E Invalid or missing message number [RC=24]
 DMSMGX408E Number of substitutions exceeds 20 [RC=24]
 DMSMGX631E XMITMSG must be invoked from an EXEC 2 or REXX exec or
 as a CMS command [RC=24]

Additional system messages may be issued by this command. The reasons for these
 messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

Immediate Commands

You can issue an Immediate command from the terminal only after causing an attention interruption by pressing the Attention key (or its equivalent, such as the Enter key). These commands are processed as soon as they are entered. The HT and RT Immediate commands are also recognized when they are stacked in an exec procedure, and the HT Immediate command can be appended to a CMS command preceded by a logical line end symbol (#). Any program execution in progress is suspended until the Immediate command is processed.

None of the Immediate commands issue responses.

The Immediate commands are:

- HB - Halt batch execution
- HI - Halt interpretation
- HO - Halt tracing
- HT - Halt typing
- HX - Halt execution
- RO - Resume tracing
- RT - Resume typing
- SO - Suspend tracing
- TE - Trace end
- TS - Trace start

You can define your own Immediate commands by using any of the following:

- the IMMCMD macro in an assembler language program.
- the IMMCMD command within an exec (CMS EXEC, EXEC 2, System Product Interpreter).
- NUCXLOAD command with the IMMCMD option specified.

For a general discussion on Immediate commands, see the *VM/SP CMS User's Guide*.

HB

Use the HB (Halt batch execution) command to stop the execution of a CMS batch virtual machine at the end of the current job.

Format

HB	
-----------	--

Usage Notes

1. If the batch virtual machine is running disconnected, it must be reconnected.
2. When the HB command is executed, CMS sets a flag such that at the end of the current job, the batch processor generates accounting information for the current job and then logs off the CMS batch virtual machine.

HI

Use the HI (Halt Interpretation) command to cause all currently executing System Product Interpreter or EXEC 2 programs or macros to terminate execution without destroying the environment (as HX would).

Format

HI	
-----------	--

HO

Use the HO (Halt tracing) command during the execution of a command or one of your programs to stop the recording of trace information. Program execution continues to its normal completion, and all recorded trace information is spooled to the printer.

Format

HO	
-----------	--

HT

Use the HT (Halt typing) command to suppress all terminal output generated by any CMS command or your program that is currently executing.

Format

HT	
-----------	--

Usage Notes

1. Program execution continues. When the ready message is displayed, normal terminal output resumes. Use the RT command to restore typing or displaying.
2. CMS error messages having a suffix letter of S or T cannot be suppressed.
3. When using the LINEDIT macro with the DISP=SIO option, output to the screen will not be suppressed. This is also true when the CONSOLE macro is used to write text to the console.
4. When using full-screen CMS, HT purges any nonpriority output that is in the queue for the virtual screen to which message class CMS is routed. (For more information on message class routing, see the ROUTE command). All priority output is displayed, such as messages and warnings through IUCV, ECHO output, and output from applications that use the LINEWRIT macro coded with PRIOR=YES.

HX

Use the HX (Halt execution) command to stop the execution of any CMS or CMS/DOS command or program, close any open files or I/O devices, and return to the CMS command environment.

Format

HX	
----	--

Usage Notes

1. HX clears all file definitions made via the FILEDEF or DLBL commands, including those entered with the PERM option.
2. HX clears all namedefs that you created with the CREATE NAMEDEF command.
3. The HX command is executed when the next SVC or I/O interruption occurs: therefore a delay may occur between keying HX and the return to CMS. All terminal output generated before HX is processed is displayed before the command is executed.
4. HX causes all storage of type 'user' to be released.
5. If you issue HX while using full-screen CMS, data is not logged in a LOGFILE (see SET LOGFILE) for the command or program that is executing.

RO

Use the RO (Resume tracing) command, during the execution of a command or one of your programs, to resume the recording of trace information that was temporarily suspended by the SO command. Program execution continues to its normal completion, and all recorded trace information is spooled to the printer.

Format

RO	
----	--

RT

Use the RT (Resume typing) command to restore terminal output from an executing CMS command or one of your programs that was previously suppressed by the HT command.

Format

RT	
----	--

Usage Note

Program execution continues, and displaying continues from the current point of execution in the program. Any terminal output that is generated after the HT command is issued and up to the time the RT command is issued is lost. Execution continues to normal program completion.

SO

Use the SO (Suspend tracing) command during the execution of a command or one of your programs to temporarily suspend the recording of trace information. Program execution continues to its normal completion and all recorded trace information is spooled to the printer.

Format

SO	
----	--

Usage Note

To resume tracing, issue the RO command.

TE

Use the TE (Trace End) command to stop all tracing of your System Product Interpreter or EXEC 2 program or macro.

Format

TE	
----	--

TS

Use the TS (Trace Start) command to start tracing of your System Product Interpreter or EXEC 2 program or macro.

Format

TS	
----	--

Chapter 3. CMS Commands for Windowing

This chapter contains descriptions of CMS commands for creating and managing windows and virtual screens. At the end of this chapter are descriptions of Border commands for manipulating windows. These are single-character commands for using in the corners of a window. For general usage information regarding windowing and full-screen CMS, see the *VM/SP CMS User's Guide*.

Windows and Virtual Screens

Windowing support consists primarily of the following:

- Windows
- Virtual Screens

A window is an area on the physical screen where virtual screen data can be displayed and manipulated. A window let's you see what is in a virtual screen.

A virtual screen (vscreen for short) can be thought of as a “presentation space” where data can be stored. This space simulates a physical screen, but is not confined to your physical screen size.

When you enter input or view output through a window, you are actually working with virtual screen data. Depending upon the size of the window and the size of the virtual screen, you may see all of the virtual screen at once, or only a portion it. As a pre-requisite to using the windowing facility, you should first review the tutorial information in the *VM/SP CMS User's Guide*.

Because a window reflects a virtual screen, you may do several operations against a virtual screen and view the results in a window. The characteristics of virtual screens that you can manipulate include:

- Reserved areas for information such as titles and PF key descriptions.
- Color and highlighting
- Options to log data into a file.

You may position a window almost anywhere on the physical screen that you prefer. You may also display as many windows on the screen at a time as you want. For displaying a large number of windows, you may choose to overlap windows or display windows on top of each other. To view the data in the window, you may scroll forward, backward, right, or left through the data.

Windows are maintained in an ordered list. You may shuffle the order by “popping” and “dropping” the windows.

Using Full-Screen CMS

Full-screen capability for CMS is optional for 3270-type terminals. You may already be familiar with full-screen if you use a VM/SP editor such as XEDIT.

To request full-screen capability, issue:

```
SET FULLSCREEN ON
```


or, you may put this command in your PROFILE exec.

In full-screen CMS, you may enter a command from anywhere on the screen. You also have the capability to scroll forward and backward through your CMS session to view commands that you entered previously and to view the CMS responses to these commands. To reenter any command, you do not need to retype the entire command. Instead, you can scroll back to where the command was previously entered, retype any letter, and press ENTER.

With full-screen CMS, default virtual screens and default windows have been defined for you, and automatically route VM output and messages into windows. You may choose to control or tailor some of these features. See the *VM/SP CMS User's Guide* for more information about tailoring and customizing.

Windowing and Virtual Screen Commands

The two tables below alphabetically list the CMS commands for working with windows and virtual screens:

Table 18 (Page 1 of 2). CMS Commands for windows and virtual screens	
Command	Usage
ALARM VSCREEN	Sounds the terminal alarm the next time the display is refreshed.
CLEAR VSCREEN	Erases data in the virtual screen by overwriting the data buffer with nulls.
CLEAR WINDOW	Scrolls past all data in the virtual screen to which the window is connected so that no data is displayed in the data area of the window.
CURSOR VSCREEN	Positions the cursor on specified line and column in a virtual screen.
DEFINE VSCREEN	Creates a virtual screen.
DEFINE WINDOW	Creates a window.
DELETE VSCREEN	Removes a virtual screen definition.
DELETE WINDOW	Removes a window definition.
DROP WINDOW	Moves a window down in the order of displayed windows.
GET VSCREEN	Writes data from a CMS file to the specified virtual screen.
HIDE WINDOW	Prevents the specified window from being displayed, and connects the window to a virtual screen.
MAXIMIZE WINDOW	Expands a window to the physical screen size.
MINIMIZE WINDOW	Reduces the size of the window to one line.
POP WINDOW	Moves a window up in the order of displayed windows.
POSITION WINDOW	Changes the location of a window on the physical screen.
PUT SCREEN	Makes a copy of the physical screen and writes the image to a CMS file.
PUT VSCREEN	Writes the data from the data area of a virtual screen to a CMS file.

Table 18 (Page 2 of 2). CMS Commands for windows and virtual screens	
Command	Usage
QUERY	The following QUERY command options are used for windowing: APL, BORDER, CHARMODE, CMSPF, CURSOR, DISPLAY, FULLREAD, FULLSCREEN, HIDE, KEY, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, ROUTE, SHOW, TEXT, VSCREEN, WINDOW, and WMPF.
REFRESH	Updates virtual screens and their associated windows and refreshes the screen.
RESTORE WINDOW	Returns a maximized or minimized window to its size and location before the maximize or minimize.
ROUTE	Directs data of a particular message class to a virtual screen.
SCROLL	Moves a window to a new location on the virtual screen.
SET	The following SET command options are used for windowing: APL, BORDER, CHARMODE, CMSPF, FULLREAD, FULLSCREEN, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, TEXT, VSCREEN, WINDOW, and WMPF.
SHOW WINDOW	Places a window on top of all other displayed windows and connect the window to a virtual screen.
SIZE WINDOW	Changes the number of lines and columns for a specified window.
WAITREAD VSCREEN	Uses from an exec to update the virtual screen with data, refreshes the physical screen, and waits for the next attention interrupt.
WAITT VSCREEN	Updates the virtual screen with data.
WRITE VSCREEN	Enters information in a virtual screen.

Table 19 (Page 1 of 2). CMS Border Commands for Windows	
Command	Usage
B	Scrolls the window backward
C	Clears the window of data
D	Drops the window
F	Scrolls the window forward
H	Hides the window
L	Scrolls the window to the left
M	Changes the location of the window
N	Minimizes the window
O	Restores the window
P	Pops the window
R	Scrolls the window to the right

CMS Commands for Windowing

Table 19 (Page 2 of 2). CMS Border Commands for Windows	
Command	Usage
S	Changes the size of the window
X	Maximizes the window

ALARM VSCREEN

Use the ALARM VSCREEN command to sound the terminal alarm the next time a virtual screen is displayed in a window on the screen.

Format

ALARM VScreen	<i>vname</i>
----------------------	--------------

Operands

vname

is the name of the virtual screen for which the alarm sounds.

Responses

The terminal alarm sounds.

Messages and Return Codes

DMSALA386E Missing operand(s) [RC=24]
 DMSALA388E Invalid keyword: *keyword* [RC=24]
 DMSALA391E Unexpected operand(s): *operand* [RC=24]
 DMSALA921E Virtual screen *vscreen* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

CLEAR VSCREEN

Use the CLEAR VSCREEN command to erase data in a virtual screen.

Format

CLEAR VScreen	<i>vname</i>
----------------------	--------------

Operands

vname
is the name of the virtual screen to be cleared.

Usage Notes

- The CLEAR VSCREEN command
 - Clears the scrollable data area by overwriting the data buffer with nulls.
 - Purges data in the queue.
 - Reconnects all windows connected to the virtual screen to the top of the virtual screen.
 - Leaves the reserved areas and cursor position unchanged.
- The field attribute buffer and the extended attribute buffers for color, extended highlighting (exthi) and Programmed Symbol (PS) sets are reset to the default attributes as specified on the DEFINE VSCREEN and SET VSCREEN commands.
- When a virtual screen is cleared, the lines in the scrollable data area are renumbered if necessary. Because of this, the cursor line number returned when QUERY CURSOR *vname* is issued immediately before CLEAR VSCREEN may be different from the cursor line number returned when QUERY CURSOR *vname* is issued immediately following CLEAR VSCREEN. The physical location of the cursor in the virtual screen remains unchanged.

Messages and Return Codes

DMSCLR386E Missing operand(s) [RC=24]
 DMSCLR388E Invalid keyword: *keyword* [RC=24]
 DMSCLR391E Unexpected operand(s): *operand* [RC=24]
 DMSCLR921E Virtual screen *vname* is not defined [RC=28]
 DMSCLR928E Command is not valid for virtual screen *vname* [RC=12]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

CLEAR WINDOW

Use the CLEAR WINDOW command to scroll past all data in the virtual screen to which the window is connected so that no scrollable data is displayed in the window.

Format

CLEAR WINDOW	$\left[\begin{array}{c} wname \\ \underline{\quad} \end{array} \right]$
---------------------	--

Operands

wname

is the name of the window that is cleared. An "=" indicates that the topmost window is cleared. If this operand is not specified, "=" is assumed.

Usage Notes

1. CLEAR WINDOW is valid only when the window is connected to a virtual screen (see HIDE WINDOW and SHOW WINDOW).
2. If the window you want to clear is variable size, the window is not displayed when the physical screen is refreshed.
3. If you are writing output (see the WRITE VSCREEN command), CLEAR WINDOW provides a means for starting output at the top of the window. It does not erase data in the virtual screen but instead works like scrolling. CLEAR WINDOW positions the window at the line following the last data line in the virtual screen. When output is written, lines of data are appended to the virtual screen and are displayed starting at the first nonreserved line of the window.
4. In a System Product Editor session, you cannot clear the window that the System Product Editor is using.

Messages and Return Codes

DMSCLR386E	Missing operand(s) [RC = 24]
DMSCLR388E	Invalid keyword: <i>keyword</i> [RC = 24]
DMSCLR391E	Unexpected operand(s): <i>operand</i> [RC = 24]
DMSCLR921E	Window <i>wname</i> is not defined [RC = 28]
DMSCLR929E	Window <i>wname</i> is not connected to a virtual screen [RC = 36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

CURSOR VSCREEN

Use the CURSOR VSCREEN command to position the cursor on a specified line and column in a virtual screen.

Format

CURsor VSCreen	<i>vname line col</i> [(options [)]]
	<u>Options:</u> [<u>Reserved</u> <u>Data</u>]

Operands

vname
is the name of the virtual screen.

line
is the line number in the virtual screen where the cursor is positioned.

col
is the column in the virtual screen where the cursor is positioned.

Options

Reserved

places the cursor in the reserved area of the virtual screen.

The *line* number must be less than or equal to the number of lines in the reserved area. A negative line number positions the cursor in the bottom reserved area, and a positive line number positions the cursor in the top reserved area.

The *col* must be less than or equal to the number of columns in the virtual screen.

You cannot specify a *line* or *col* of zero when placing the cursor in the RESERVED area.

Data

places the cursor in the scrollable data area at the specified line and column. DATA is the default.

The *line* number must be less than or equal to the number of lines in the data area, and must be zero or greater. A value of zero positions the cursor at the line following the current bottom of the virtual screen.

The *col* must be less than or equal to the number of columns in the virtual screen, and must be zero or greater. A value of zero positions the cursor in column two. This allows for a start field in column one.

Usage Notes

1. In a virtual screen, the lines in the top reserved area are numbered starting from the top. The top line is 1, the second line is 2, etc. In the bottom reserved area, lines are numbered starting at the bottom and have negative values. The bottom line is line -1, the second line up is -2, the third line up is -3, etc.
2. When the physical screen is read, the following cursor information is saved:
 - a. Cursor position on the physical screen
 - b. Cursor position in the window and the window name
 - c. Cursor position in the virtual screen and the virtual screen name.

If the cursor is not in a window or if it is on a border, the window and virtual screen cursor information is not updated.

In addition, the cursor position can be changed by:

- The CURSOR VSCREEN command
- Lines written to a virtual screen

The last window and last virtual screen to be updated with cursor location information are said to 'own' the cursor.

When the physical screen is refreshed, the following rules determine the position of the cursor on the physical screen:

- a. If a border command (with the exception of X and H) was issued, position the cursor on the corner of the window in which the last processed border command was typed. If the window position is no longer displayed, processing continues with the next step.
- b. Place the cursor in the virtual screen that last owned the cursor:
 - 1) Determine if the last window to own the cursor is connected to this virtual screen. If it is, check if the window contains the virtual screen cursor position and if it is visible on the physical screen.
 - 2) If the position is not visible and this is the CMS virtual screen, place the cursor on the command line.
 - 3) Otherwise, check other windows connected to this virtual screen and determine if any of these windows contain the virtual screen cursor position and are visible on the physical screen. Place cursor in the first window that meets these criteria.
 - 4) For each window connected to the virtual screen that owns the cursor, starting with the top-most window, determine if the virtual screen displayed by that window has a visible cursor position. Place the cursor in the first position visible on the physical screen.
 - 5) Calculate the cursor positions of each window visible on the physical screen (starting with the top-most window) and place the cursor in the first position that is visible on the physical screen.
- c. Place the cursor in the window that last owned the cursor:
 - 1) Place the cursor in the window that last owned it if that position is visible on the physical screen.
 - 2) If the position not visible, determine if any position in this window is visible on the screen and place it in the first position of this window that is visible.

CURSOR VSCREEN

- 3) Calculate the cursor position for the top window. Place the cursor in this position if it is visible. Otherwise, place it in the first visible position of the top window.
- d. If the cursor still has not been positioned, place it in row 1, column 1 of the physical screen.

Responses

The next time the screen is displayed, the cursor will be positioned at the specified location in the virtual screen, assuming that the location is displayed somewhere in an associated window on the physical screen.

Messages and Return Codes

DMSCUR386E Missing operand(s) [RC=24]
DMSCUR388E Invalid keyword: *keyword* [RC=24]
DMSCUR389E Invalid operand: *operand* [RC=24]
DMSCUR391E Unexpected operand(s): *operand* [RC=24]
DMSCUR394E Invalid option: *option* [RC=24]
DMSCUR921E Virtual screen *vname* is not defined [RC=28]
DMSCUR923E Specified location is outside the virtual screen [RC=32]
DMSCUR928E Command is not valid for virtual screen *vname* [RC=12]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

DEFINE VSCREEN

Use the DEFINE VSCREEN command to create a virtual screen. A virtual screen is a functional simulation of a physical display screen. It is a “presentation space” where data is written.

Format

DEFine VScreen	<p><i>vname lines cols rtop rbot</i> [(optionA optionB optionC optionD)]]</p> <p>OptionA: [<u>TYPe</u> [<u>NOType</u>]]</p> <p>OptionB: [<u>PRotect</u> [<u>High</u> [<u>NOProtect</u>] [<u>NOHigh</u>]]]</p> <p>OptionC: [<i>color</i>] [<i>exthi</i>] [<i>psset</i>]</p> <p>OptionD: [<u>USer</u> [<u>SYstem</u>]]</p>
-----------------------	--

Operands

vname

is the name assigned to the virtual screen. You may specify a name up to eight characters in length.

lines

is the number of scrollable data lines that the virtual screen contains. The number of lines must be one or greater.

cols

is the number of columns that the virtual screen contains.

rtop

is the number of reserved lines maintained in the top reserved area of the virtual screen.

rbot

is the number of reserved lines maintained in the bottom reserved area of the virtual screen.

Option A

Option A controls the actions when information is written to the specified virtual screen.

The following may be specified:

TYPe

specifies that data is moved to the virtual screen when the virtual screen queue is processed. TYPE is the default.

NOType

specifies that the virtual screen is not updated.

DEFINE VSCREEN

Option B

Option B indicates the default attributes of the data in the virtual screen. The following may be specified:

PRotect

the data is protected.

NOPRotect

the data is not protected. NOPRotect is the default.

High

data is displayed in high intensity.

NOHigh

data is displayed in a normal intensity. NOHigh is the default.

Option C

Option C indicates the default extended attributes for the virtual screen. The following may be specified:

color

the color may be Default, Blue, Red, Pink, Green, Turquoise, Yellow, or White.

exthi

the extended highlighting may be None (default), REVvideo, BLInk, or Underline.

psset

the Programmed Symbol Set (PSset) may be specified as PS0 (the default), PS1, PSA, PSB, PSC, PSD, PSE, or PSF.

Option D

Option D indicates whether or not the virtual screen is retained when a task abnormally ends (abend) or when the HX (halt execution) command is issued. The following may be specified:

USer

indicates that the virtual screen is deleted when a task abnormally ends (abend) or when the HX (halt execution) command is issued.

SYstem

indicates that the virtual screen is retained when a task abnormally ends (abend) or when the HX (halt execution) command is issued.

Usage Notes

1. For information on attributes, extended attributes, fields, and Programmed Symbol Sets, refer to the *IBM 3270 Information Display System Data Stream Programmer's Reference, GA23-0059*.
2. Use the SET VSCREEN command to change the options for a defined virtual screen.
3. When you specify NOTYPE, the virtual screen is not updated when the queue is processed. However, the data in the queue is logged to a CMS file if logging is set on (see the SET LOGFILE command).
4. The virtual screen options are accepted whether or not the device has the ability to use those options. However, the action taken depends upon the device. For example, color is ignored on a 3278 display, and color and extended highlighting are ignored on a 3277 display. In addition, if Programmed Symbol Sets are

supported by the device, then the PSset specified must be loaded in the display. If not, the default PSset for the device is used.

QUERY VSCREEN displays all the options, even those that may be ignored. The QUERY DISPLAY command displays the options that are supported by the device.

5. When you define a virtual screen, the following buffers are allocated for the data and option information:

- Data
- Attribute
- Color
- Extended highlight
- PSset

The number of buffers allocated depends on the options supported by the device. For example, if color is not available, a color buffer is not allocated. The data buffer and the attribute buffer are always defined.

The structure allows you to assign different characteristics to each character in a virtual screen. For example, adjacent characters can be displayed with different colors if the device supports character options. The SET CHARMODE command allows you to specify whether character attributes should be used when displaying virtual screen data.

In addition to the buffers, a queue is also defined. Data is queued to the virtual screen when writes are done.

6. The option information sets the default display characteristics of the data in the virtual screen. When data is written to the virtual screen, the virtual screen defaults are used unless the write specifies different characteristics, in which case the write options override the characteristics of the virtual screen. See the WRITE VSCREEN command for more information.

7. The reserved lines are maintained outside the area defined as scrollable data in the virtual screen. For example:

```
define vscreen message 20 80 1 1
```

defines a virtual screen named MESSAGE that contains 20 lines of scrollable data, one line in the top reserved area, and one line in the bottom reserved area. Each line is 80 columns wide. The WRITE VSCREEN command enables you to write to the scrollable data area or, using the RESERVED option, write to the reserved areas.

8. A virtual screen may not be defined with the same name as a virtual screen that already exists. The rules for naming a virtual screen are the same as those for naming files:

- The name can be from one to eight characters.
- The valid characters are A-Z, a-z, 0-9, \$, #, @, +, - (hyphen), :(colon), and _ (underscore).

9. To view the contents of a virtual screen in a window, a window must be connected to the virtual screen using the SHOW WINDOW or HIDE WINDOW command.

DEFINE VSCREEN

Messages and Return Codes

DMSDEF014E Invalid function *function* [RC=24]
DMSDEF386E Missing operand(s) [RC=24]
DMSDEF389E Invalid operand: *operand* [RC=24]
DMSDEF391E Unexpected operand(s): *operand* [RC=24]
DMSDEF394E Invalid option: *option* [RC=24]
DMSDEF622E Insufficient free storage [RC=104]
DMSDEF913E Invalid virtual screen name: *vname* [RC=20]
DMSDEF920E Virtual screen *vname* already exists [RC=3]
DMSDEF926E Command is only valid on a display terminal [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

DEFINE WINDOW

Use the DEFINE WINDOW command to create a window with the specified name, size, and position on the physical screen.

Format

DEFine WINDOW	<i>wname lines cols psline pscol [(options [])]</i> Options: [<u>V</u> ARiable] [<u>B</u> ORder] [<u>P</u> OP] [<u>F</u> IXed] [<u>N</u> OBorder] [<u>N</u> OPop] [<u>T</u> OP] [<u>U</u> Ser] [<u>N</u> OTop] [<u>S</u> Ystem]
----------------------	---

Operands

wname

is the name assigned to the window. You may specify a name up to eight characters in length.

lines

is the number of lines the window displays.

cols

is the number of columns the window displays.

psline

is the line on the physical screen where the upper or lower edge of the window is positioned. A positive number positions the upper edge of the window on the specified line relative to the top of the screen. A negative number positions the lower edge of the window on the specified line relative to the bottom of the screen. For more information, see Usage Note 7.

pscol

is the column on the physical screen where the left edge of the window is positioned.

Options

VARiable

indicates that the number of lines in the window may vary depending on the amount of scrollable data displayed.

FIXed

indicates that the number of lines in the window is always constant. FIXED is the default.

BORder

indicates that the borders are displayed when possible. BORDER is the default.

NOBorder

indicates that borders are not displayed.

DEFINE WINDOW

POP

specifies that the window is displayed on top of all other windows when the virtual screen that the window is showing is updated.

NOPOP

specifies that there is no effect on the window's position in the ordered list of windows when the virtual screen that the window is showing is updated.

NOPOP is the default.

TOP

specifies that the window may qualify as the topmost window. Most windowing commands process the topmost window by default or when = is specified as the window name.

NOTop

specifies that the window cannot qualify as the topmost window. Windows defined as NOTOP are not processed by default or when a command is specified with = for the window name.

USer

indicates that the window is deleted when a task abnormally ends (abend) or when the HX (halt execution) command is issued.

SYstem

indicates that the window is retained when a task abnormally ends (abend) or when the HX (halt execution) command is issued.

Usage Notes

1. A maximum of 255 windows may be defined at any time.
2. A window may not be defined with the same name as a window that already exists. A window name of "*" and "=" is invalid.
3. Defining a window does not automatically display it. To display a window, it must be connected to a virtual screen (see the SHOW WINDOW and HIDE WINDOW commands for details).
4. Use the SET WINDOW command to change the options for a window.
5. A window displays as many top and bottom reserved lines as possible, regardless of the position on the virtual screen where the window is connected. The reserved lines are defined and maintained in the virtual screen. See the SET RESERVED command to change the way in which reserved lines are displayed in a window.
6. A window's size and location must be specified in such a way that, excluding borders, the entire window fits on the physical screen.
7. Pslines may be specified as either a positive or a negative number. When positive, the window's upper left corner is placed on the physical screen at the line number specified. When negative, the window's lower left corner is placed relative to the bottom of the physical screen. Thus, a psline value of +1 positions a window by its upper left corner to the top line of the physical screen. A psline value of -1 positions it by its lower left corner to the bottom line of the physical screen.
8. Window borders are built outside the area defined for the window. Therefore, it is possible that some or all of the borders may not fit on the physical screen due to the size and position of the window. Borders are highlighted and displayed using the following characters:

- top border character is a dash, '-'
- bottom border character is a dash, '-'
- left border character is a vertical bar, '|'
- right border character is a vertical bar, '|'

Border corners are identified with a plus (+) sign. Use the SET BORDER command to alter border attributes and characters.

9. Single-character Border commands can be entered in the border corners. See the "Border Commands" on page 778 and the *VM/SP CMS User's Guide* for more information on the border commands.
10. If the window, virtual screen, and physical screen do not have the same number of columns, it is recommended that you define the window with one column greater than the number of columns in the virtual screen that it is displaying. This provides for the additional field definition character (Start Field) that is necessary for the proper display of the window on the screen, and ensures that a maximum number of columns of virtual screen data are displayed.
11. When you specify a window as VARIABLE, the current number of lines in the window may vary from 0, in which case the window is not displayed, to the number of lines specified for the window. The window size depends upon the amount of scrollable data being displayed.
12. If multiple windows defined with the POP option are showing the same virtual screen, all the windows are displayed on top of the other windows when the virtual screen is updated. Their position in relation to each other is maintained.

Messages and Return Codes

- DMSDEF014E Invalid function *function* [RC=24]
- DMSDEF386E Missing operand(s) [RC=24]
- DMSDEF389E Invalid operand: *operand* [RC=24]
- DMSDEF391E Unexpected operand(s): *operand* [RC=24]
- DMSDEF394E Invalid option: *option* [RC=24]
- DMSDEF622E Insufficient free storage [RC=104]
- DMSDEF676E Invalid character {*|=} for window name [RC=20]
- DMSDEF915E Maximum number of windows already defined [RC=12]
- DMSDEF920E Window *wname* already exists [RC=3]
- DMSDEF922E Window does not fit entirely on the screen [RC=32]
- DMSDEF926E Command is only valid on a display terminal [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DELETE VSCREEN

Use the DELETE VSCREEN command to remove a virtual screen definition.

Format

DELeTe VSCreen	<i>vname</i>
-----------------------	--------------

Operands

vname

is the name of the virtual screen to be deleted.

Usage Notes

1. The CMS virtual screen cannot be deleted when SET FULLSCREEN is ON or SUSPEND (see SET FULLSCREEN).
2. When deleting a virtual screen and data is in the queue, handling of data depends on whether or not you are using full-screen CMS. When SET FULLSCREEN is ON, the data is redirected to the CMS virtual screen and window. When SET FULLSCREEN is OFF or SUSPEND, the data is typed out on your screen.
3. When you delete a virtual screen, all windows connected to it are disconnected. Any ROUTE command message classes that have been directed to the virtual screen are rerouted to the CMS virtual screen.

Messages and Return Codes

DMSDEL386E Missing operand(s) [RC=24]
 DMSDEL388E Invalid keyword: *keyword* [RC=24]
 DMSDEL391E Unexpected operand(s): *operand* [RC=24]
 DMSDEL919E The CMS virtual screen cannot be deleted [RC=24]
 DMSDEL921E Virtual screen *vname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DELETE WINDOW

Use the DELETE WINDOW command to remove a window definition.

Format

DELEte WInDow	<i>wname</i>
---------------	--------------

Operands

wname

is the name of the window to be deleted.

Usage Notes

1. If the window is connected to a virtual screen, the virtual screen (vscreen) is not affected.
2. The CMS window cannot be deleted when SET FULLSCREEN is ON or SUSPEND (see SET FULLSCREEN).

Messages and Return Codes

DMSDEL386E Missing operand(s) [RC=24]
 DMSDEL388E Invalid keyword: *keyword* [RC=24]
 DMSDEL391E Unexpected operand(s): *operand* [RC=24]
 DMSDEL919E The CMS window cannot be deleted [RC=24]
 DMSDEL921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

DROP WINDOW

Use the DROP WINDOW command to move a window down in the order of displayed windows.

Format

DROP WINDOW	$\left\{ \begin{array}{c} wname \\ = \\ WM \end{array} \right\} \left[\begin{array}{c} n \\ * \\ - \end{array} \right]$
--------------------	--

Operands

wname

is the name of the window to be dropped.

=

indicates that the topmost window is dropped.

WM

hides the WM window so that it is no longer visible on the screen and commands cannot be entered from it. You are returned to the environment you were in before entering POP WINDOW WM. The *n* and * have no effect when specified with WM.

n

is the number of positions the window is moved down. An "*" positions the window behind all other windows. If this operand is not specified, "*" is assumed.

Usage Notes

1. If the window is hidden (see HIDE WINDOW), then DROP WINDOW has no effect.
2. When using full-screen CMS, you cannot drop a window that is showing the virtual screen indicated in the status area message. For example, if the CMS window is showing the CMS virtual screen, and the status area message instructs you to "Scroll for more information in vscreen CMS," you cannot drop the CMS window. You can drop any other window.
3. In the WM window, the PA2 and CLEAR keys scroll the topmost window forward. When there is no more data in the window to scroll, you automatically exit the WM environment.

Messages and Return Codes

DMSDRP386E	Missing operand(s) [RC = 24]
DMSDRP388E	Invalid keyword: <i>keyword</i> [RC = 24]
DMSDRP389E	Invalid operand: <i>operand</i> [RC = 24]
DMSDRP391E	Unexpected operand(s): <i>operand</i> [RC = 24]
DMSDRP921E	Window <i>wname</i> is not defined [RC = 28]
DMSDRP929E	Window <i>wname</i> is not connected to a virtual screen [RC = 36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

GET VSCREEN

Use the GET VSCREEN command to write lines from a CMS file to the specified virtual screen.

Format

GET VScreen	<i>vname fn ft</i> [<i>fm</i> [<i>fromrec</i> [<i>numrec</i>]]]
--------------------	---

Operands

vname

is the name of the virtual screen to be updated with the data in the specified CMS file.

fn

is the file name of the file.

ft

is the file type of the file.

fm

is the file mode of the file. If you do not specify *fm* or if * is specified, then all accessed disks and directories are searched.

fromrec

is the starting record of the file that is moved into the virtual screen. GET VSCREEN starts with the first record in the file by default.

numrec

is the number of records that are read from the file and moved into the virtual screen. All the records are read by default. An * means that records are processed until the end of the file is reached.

Usage Notes

1. GET VSCREEN queues each record of a CMS file to the virtual screen. The information is appended to the end of the virtual screen when the virtual screen is updated. See WRITE VSCREEN for more information on queues.

Messages and Return Codes

DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC=31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSGET069E	Filemode <i>mode</i> not accessed [RC=36]
DMSGET386E	Missing operand(s) [RC=24]
DMSGET388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSGET389E	Invalid operand: <i>operand</i> [RC=24]
DMSGET391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSGET921E	Virtual screen <i>vname</i> is not defined [RC=28]
DMSSTT048E	Invalid mode <i>mode</i> [RC=24]
DMSSTT062E	Invalid character <i>char</i> in fileid [<i>fn ft [fm]</i>] [RC=20]
DMSWVL002E	File <i>fn ft fm</i> not found [RC=28]
DMSWVL104S	Error <i>nn</i> reading file <i>fn ft fm</i> from disk or directory [RC=31 55 70 76 99 100]

DMSWVL109S Virtual storage capacity exceeded [RC = 104]
DMSWVL156E Record *nnn* not found -- file *fn ft fm* has only *nnn* records
[RC = 32]
DMSWVL928E Command is not valid for virtual screen *vname* [RC = 12]
DMSWVL1262S Error *nn* opening file *fn ft fm* [RC = 31|55|70|76|99|100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

HIDE WINDOW

Use the HIDE WINDOW command to prevent the specified window from being displayed and to connect the window to a virtual screen. The SHOW WINDOW command will redisplay the window.

Format

HIDE WINDOW	[<u>wname</u> [ON vname [line col]]]
--------------------	---

Operands

wname

is the name of the window to be hidden. An "=" indicates that the topmost window is hidden. If wname is not specified, "=" is assumed.

vname

is the name of the virtual screen to which the window will be connected.

line

is the virtual screen line number where the upper left corner of the window is placed.

col

is the column number of the virtual screen where the upper left corner of the window is placed.

Usage Notes

1. Multiple windows may be connected to a single virtual screen.
2. If the window is already connected to a virtual screen when you issue the HIDE WINDOW command, you do not have to specify the virtual screen information.
3. When you specify a virtual screen name, line, and column, the line and column values must be less than or equal to the corresponding virtual screen dimensions. The minimum line and column value is 1. If line and column are not specified, the default is 1 for both.

If the specified line is past the current bottom of the virtual screen, the window is connected to the bottom of the virtual screen.

4. When you hide a window, the window is removed from the list of displayed windows. Therefore, issuing a command like POP WINDOW or DROP WINDOW that manipulates the order of displayed windows has no effect on hidden windows.
5. You must always have a window showing the CMS virtual screen when using full-screen CMS. If you hide all the windows showing the CMS virtual screen, the CMS window is automatically shown at the top of the CMS virtual screen. The CMS window is on top of all other windows, including the STATUS window. You should then issue the POP WINDOW STATUS command.

Messages and Return Codes

DMSHID386E Missing operand(s) [RC=24]
 DMSHID388E Invalid keyword: *keyword* [RC=24]
 DMSHID389E Invalid operand: *operand* [RC=24]
 DMSHID391E Unexpected operand(s): *operand* [RC=24]
 DMSHID921E Window *wname* is not defined [RC=28]
 DMSWMM921E Virtual screen *vname* is not defined [RC=28]
 DMSWMM921E Window *wname* is not defined [RC=28]
 DMSWMM923E Specified location is outside the virtual screen [RC=32]
 DMSWMM929E Window *wname* is not connected to a virtual screen [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

MAXIMIZE WINDOW

Use the MAXIMIZE WINDOW command to expand a window to the physical screen size.

Format

MAXimize Window	[<i>wname</i>] =
------------------------	-----------------------

Operands

wname

is the name of the window to be expanded. An "=" indicates that the topmost window is expanded. If this operand is not specified, "=" is assumed.

Usage Notes

1. The RESTORE WINDOW command returns the window to its size and location prior to the maximize.
2. A window can be maximized whether or not it is connected to a virtual screen and whether or not the window is displayed.
3. A maximized window is positioned at line 1, column 1 of the physical screen.
4. A variable size window that is maximized still retains its variable size properties. Thus, depending on how many lines exist in the virtual screen to which the window is connected, the window may appear to be less than full screen size when it is displayed on the physical screen.

For example, suppose a variable size window is connected to line one of a virtual screen that contains three data lines. When the window is maximized:

- it moves to line 1, column 1;
 - its width is the width of the physical screen; and,
 - it contains only three data lines.
5. When you maximize a window so that it fills the entire screen and covers all other windows, you may not be able to enter commands. The WM window is automatically displayed, and the WMPF keys and command line are available to manipulate the window. Use the RESTORE WINDOW command to restore the window and the DROP WINDOW command to exit the WM environment.

Messages and Return Codes

DMSMAX386E Missing operand(s) [RC=24]
 DMSMAX388E Invalid keyword: *keyword* [RC=24]
 DMSMAX391E Unexpected operand(s): *operand* [RC=24]
 DMSMAX921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

MINIMIZE WINDOW

MINIMIZE WINDOW

Use the MINIMIZE WINDOW command to reduce the size of the window to one line.

Format

MINimize WInDow	[<i>wname</i>] =
-----------------	-----------------------

Operands

wname

is the name of the window to be reduced. An "=" indicates that the topmost window will be reduced. If this operand is not specified, "=" is assumed.

Usage Notes

1. The RESTORE WINDOW command returns the window to its size and location prior to issuing the MINIMIZE command.

Messages and Return Codes

DMSMIN386E Missing operand(s) [RC=24]
DMSMIN388E Invalid keyword: *keyword* [RC=24]
DMSMIN391E Unexpected operand(s): *operand* [RC=24]
DMSMIN921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

POP WINDOW

Use the POP WINDOW command to move a window up in the order of displayed windows.

Format

POP WINDOW	$\left\{ \begin{array}{l} wname \\ WM \end{array} \right\} \left[\begin{array}{l} n \\ * \end{array} \right]$
-------------------	--

Operands

wname

is the name of the window to be moved up.

WM

displays the WM window, allowing you to enter commands from the command line or with WMPF keys. The *n* and * have no effect when specified with WM. See Usage Note 5 for a list of commands that you can enter in the WM environment.

n

is the number of positions the window is to be moved up. An "*" indicates that the window will be positioned on top of the other displayed windows. If this operand is not specified, "*" is assumed.

Usage Notes

1. If the window is hidden (see HIDE WINDOW) then POP WINDOW has no effect.
2. A variable size window is only displayed when there is at least one scrollable line to show. If a variable size window is showing a virtual screen that does not contain any scrollable lines, the POP WINDOW command for that window moves the window up in the display order, but it is not displayed.
3. When a variable size window has been cleared (using the CLEAR WINDOW command or by scrolling forward), the POP WINDOW command moves the window up in the order of displayed windows and scrolls it to the bottom of the virtual screen that it is showing.
4. When you are using full-screen CMS and you enter the POP WINDOW command from the command line, the command is executed and then the screen is refreshed. As part of the refresh processing, any pop-type window that has output waiting is moved to the top of the order of displayed windows. Therefore, the window that you specified may not be displayed at the top of the display order because another has been popped afterwards.
5. You can display the WM window whether or not you are using full-screen CMS, that is, whether SET FULLSCREEN is ON, OFF, or SUSPEND. From the WM window you can issue the following commands:

POP WINDOW

CLEAR WINDOW	PUT SCREEN	SCROLL
CP	QUERY BORDER	SET BORDER
DROP WINDOW	QUERY HIDE	SET LOCATION
HELP	QUERY LOCATION	SET RESERVED
HIDE WINDOW	QUERY RESERVED	SET WINDOW
MAXIMIZE WINDOW	QUERY SHOW	SET WMPF
MINIMIZE WINDOW	QUERY WINDOW	SHOW WINDOW
POP WINDOW	QUERY WMPF	SIZE WINDOW
POSITION WINDOW	RESTORE WINDOW	

In the WM environment, you can enter HELP (WMPF 1) to see the list of commands that are available. The WM environment creates a WMHELP window and WMHELP virtual screen to display the list. To exit the WM window, use the DROP WINDOW command.

6. The POP WINDOW WM command automatically defines the WM window and WM virtual screen if they do not already exist.
7. You may encounter a situation where the entire screen is protected and you are unable to enter commands. For example, you may maximize a window so that it fills the entire screen and covers all other windows. You may not be able to enter commands in the window because it is protected. In such cases, the WM window is automatically displayed, and the WMPF keys and command line are available to manipulate the window. For example, use the POP WINDOW command to change the order of displayed windows or the RESTORE WINDOW command to restore a maximized window. The DROP WINDOW WM command (or the default WMPF 3 key) exits the WM environment and displays all the windows that are showing the virtual screen that is waiting for a response. If that virtual screen is the CMS virtual screen, the STATUS window is also displayed. Also, you can use the PA2 and CLEAR keys to scroll the topmost window forward. When there is no more data to scroll, you automatically exit the WM window.

Messages and Return Codes

DMSPP0386E	Missing operand(s) [RC=24]
DMSPP0388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSPP0389E	Invalid operand: <i>operand</i> [RC=24]
DMSPP0391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSPP0921E	Window <i>wname</i> is not defined [RC=28]
DMSPP0929E	Window <i>wname</i> is not connected to a virtual screen [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

POSITION WINDOW

Use the POSITION WINDOW command to change the location of a window on the physical screen.

Format

POSition WINDOW	$\left\{ \begin{array}{l} wname \\ = \end{array} \right\} \quad psline \quad pscol$
------------------------	---

Operands

wname

is the name of the window to be moved. An "=" indicates that the topmost window is moved.

psline

is the line on the physical screen where the upper or lower edge of the window is positioned. A positive number positions the upper edge of the window on the specified line relative to the top of the screen. A negative number positions the lower edge of the window on the specified line relative to the bottom of the screen.

pscol

is the column on the physical screen where the left edge of the window is positioned.

Usage Notes

1. The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.
2. 'Psline' may be specified as either a positive or a negative number. When positive, the window's upper left corner is placed on the physical screen at the line and column number specified. When negative, the window's lower left corner is placed relative to the bottom of the physical screen. Thus, a psline value of +1 positions a window by its upper left corner to the top line of the physical screen. A psline value of -1 positions it by its lower left corner to the bottom line of the physical screen.

Messages and Return Codes

DMSPT386E	Missing operand(s) [RC=24]
DMSPT388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSPT389E	Invalid operand: <i>operand</i> [RC=24]
DMSPT391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSPT921E	Window <i>wname</i> is not defined [RC=28]
DMSPT922E	Window does not fit entirely on the screen [RC=32]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

PUT SCREEN

Use the PUT SCREEN command to copy the image of a physical screen to a CMS file.

Format

PUT SCREEN	<i>fn</i> <i>ft</i> $\left[\begin{array}{c} \textit{fm} \\ * \\ \textit{A1} \end{array} \right]$
-------------------	---

Operands*fn*

is the file name of the file into which the image is copied.

ft

is the file type of the file into which the image is copied.

fm

is the file mode letter of the file. The default is *, which is the first read/write disk or directory in the search order containing the specified file. See the Usage Notes for more information.

Usage Notes

1. The PUT SCREEN command copies the last physical screen image that was displayed in a window.
2. If the file does not exist, it is created on the disk or directory accessed as A and the lines are inserted. If the file already exists, the data is appended to the end of the file.
3. For a file in fixed format, each line of virtual screen that is longer than the logical record length of the file is truncated. Lines that are shorter than the logical record length are padded with blanks.

Messages and Return Codes

DMSERD107S	Disk <i>mode</i> (<i>vdev</i>) is full [RC = 100]
DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC = 31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSPUT069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSPUT386E	Missing operand(s) [RC = 24]
DMSPUT388E	Invalid keyword: <i>keyword</i> [RC = 24]
DMSPUT391E	Unexpected operand(s): <i>operand</i> [RC = 24]
DMSPUT917E	No windows are displayed [RC = 4]
DMSSTT048E	Invalid mode <i>mode</i> [RC = 24]
DMSSTT062E	Invalid character <i>char</i> in fileid [<i>fn ft [fm]</i>] [RC = 20]
DMSWVL037E	Filemode <i>mode</i> is accessed as read/only [RC = 12]
DMSWVL069E	Filemode <i>fm</i> not accessed [RC = 36]
DMSWVL105S	Error <i>nn</i> writing file <i>fn ft fm</i> on disk or directory [RC = 100]
DMSWVL531E	Disk is full; set new filemode or clear some disk space [RC = 13]
DMSWVL924E	Data was truncated [RC = 3]
DMSWVL1258E	You are not authorized to write to file <i>fn ft fm dirid</i> [RC = 28]
DMSWVL1262S	Error <i>nn</i> opening file <i>fn ft fm</i> [RC = 100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

PUT VSCREEN

Use the PUT VSCREEN command to write the data from the scrollable data area of a virtual screen to a CMS file.

Format

PUT VScreen	$vname \ fn \ ft \ \left[\begin{array}{c} fm \\ * \\ A1 \end{array} \left[\begin{array}{c} fromlin \\ 1 \end{array} \left[\begin{array}{c} numlin \\ * \end{array} \right] \right] \right]$
-------------	--

Operands

vname

is the name of the virtual screen from which the data is written.

fn

is the filename of the file into which the data is written.

ft

is the file type of the file into which the data is written.

fm

is the file mode of the file. The default is *, which is the first read/write disk or directory in the search order containing the specified file. See the Usage Notes for more information.

fromlin

is the starting line in the virtual screen to be copied into the file. PUT VSCREEN starts with the first line in the virtual screen by default. The number specified for *fromlin* must be within the range of the current top and bottom of the virtual screen.

numlin

is the number of lines to be written into the specified file. If *numlin* is not specified, the default is *, meaning all the lines in the virtual screen starting with the specified line (*fromlin*) are copied into the file.

Usage Notes

1. If the specified file does not exist, the file is created on the disk or directory accessed as A and the lines are inserted. If the file already exists, the data is appended to the end of the file.

Messages and Return Codes

DMSERD107S	Disk <i>mode</i> (<i>vdev</i>) is full [RC=100]
DMSFNS1144E	Implicit rollback occurred for work unit <i>workunitid</i> [RC=31]
DMSFNS1252T	Rollback unsuccessful for file pool <i>filepoolid</i>
DMSPUT069E	Filemode <i>mode</i> not accessed [RC=36]
DMSPUT386E	Missing operand(s) [RC=24]
DMSPUT388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSPUT389E	Invalid operand: <i>operand</i> [RC=24]
DMSPUT391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSPUT921E	Virtual screen <i>vname</i> is not defined [RC=28]
DMSSTT048E	Invalid mode <i>mode</i> [RC=24]

- DMSSTT062E Invalid character *char* in fileid [*fn ft [fm]*] [RC = 20]
- DMSWVL037E Filemode *mode* is read only [RC = 12]
- DMSWVL069E Filemode *fm* not accessed [RC = 36]
- DMSWVL105S Error *nn* writing file *fn ft fm* on disk or directory
[RC = 31|55|70|76|99|100]
- DMSWVL109S Virtual storage capacity exceeded [RC = 104]
- DMSWVL531E Disk is full; set new filemode or clear some disk space [RC = 13]
- DMSWVL923E Specified location is outside the virtual screen [RC = 32]
- DMSWVL924E Data was truncated [RC = 3]
- DMSWVL928E Command is not valid for virtual screen *vname* [RC = 12]
- DMSWVL936W Virtual screen *vname* is empty [RC = 0]
- DMSWVL1258E You are not authorized to write to file *fn ft fm|dirid* [RC = 28]
- DMSWVL1262S Error *nn* opening file *fn ft fm* [RC = 31|55|70|76|99|100]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

QUERY

QUERY

Use the **QUERY** command to gather information about your **CMS** windowing and virtual screen environment. You can determine:

- The state of virtual machine characteristics that are controlled by the **CMS SET** command
- Physical screen characteristics
- **CMSPF** and **WMPF** key settings
- Window characteristics and the order in which the windows are being displayed
- Virtual screen characteristics
- Cursor position

Query	<pre> APL BORDER <i>uname</i> [ALL] CHARMODE CMSPF [<i>nn</i> * -] CURSOR [<i>vname</i>] DISPLAY FULLREAD FULLSCREEN HIDE [<i>uname</i> * -] KEY LINEND LOCATION <i>uname</i> LOGFILE <i>vname</i> NONDISP REMOTE RESERVED <i>uname</i> ROUTE [<i>msgclass</i> * -] SHOW [<i>uname</i> * -] TEXT VSCREEN [<i>vname</i> [ALL] * -] WINDOW [<i>vname</i> [ALL] = * -] WMPF [<i>nn</i> * -] </pre>	<p>[(options ... [])]</p>
	<p>Options: [STACK [FIFO / LIFO]]</p>	

APL

displays the status of APL character code conversion.

Response:

APL ON

or

APL OFF

where:

ON

converts APL characters for windows.

OFF

does not convert APL characters.

BORDER wname

displays whether or not the window borders are ON, and for each edge that is defined, it displays the edge name and character.

Response:

```
BORDER wname ON ([TOP char] [BOTTOM char] [LEFT char] [RIGHT char]
                OFF
```

where:

wname

is the name of the window.

char

is the character assigned to the border edge.

BORDER wname ALL

displays whether or not the window borders are ON and, for each edge that is defined, displays the edge name and character. In addition, the border attributes are also displayed.

Response:

```
BORDER wname ON|OFF ( [TOP char] [BOTTOM char] [LEFT char] [RIGHT char]
attr color exthi psset
```

where:

attr

is the attribute of the border. It may be a HIGH or NOHIGH.

color

is the color of the border.

exthi

is the extended highlighting of the border.

psset

is the Programmed Symbol set of the border.

Note: When using the ALL operand, the response may be too long for one line. If so, it wraps to the next one.

CHARMODE

displays whether character attributes or field attributes are used when displaying virtual screen data on the physical screen.

Response:

```
CHARMODE ON
```

or

```
CHARMODE OFF
```

where:

ON

indicates that character attributes are used when displaying virtual screen data on the physical screen.

OFF

indicates that field attributes are used when displaying virtual screen data on the physical screen.

CMSPF nn

displays the definition of a specific CMS PF key represented by *nn*.

Responses:

CMSPF nn [pseudonym keyword string]

where:

nn

is the number of the PF key.

pseudonym

is a 9-character representation that is displayed in the PF Key definition area at the bottom of the CMS window.

keyword

indicates when the command associated with the PF key is executed in relation to other commands entered at the terminal. It may be DELAYED, ECHO, or NOECHO. A CMSPF key set to RETRIEVE does not have a keyword associated with it.

string

is the command string associated with the PF Key.

A CMSPF key that is undefined is displayed as:

CMSPF nn

CMSPF *

displays the definitions for all full-screen CMS PF keys. This is the default.

Response:

The response is the same as for QUERY CMSPF nn; one line is displayed for each PF key.

CURSOR

displays the location of the cursor on the physical screen when the screen was last read. If there is associated virtual screen information, the name of the virtual screen and the location of the cursor in the virtual screen when the screen was last read is also displayed.

Responses:

CURSOR pline pcol [IN vname vline vcol RESERVED|DATA]

where:

pline

is the line number of the physical screen.

pcol

is the column number of the physical screen.

vname

is the name of the virtual screen that last set the cursor.

QUERY

vline

is the line number of the virtual screen. A value of zero (0) indicates that the cursor is below the current bottom.

vcol

is the column number of the virtual screen.

RESERVED|DATA

is the area where the cursor is set.

CURSOR *vname*

displays the name of the virtual screen and the current location of the cursor in the virtual screen. If the cursor has not yet been set, *vline* and *vcol* are -1.

Response:

```
VSCREEN vname vline vcol [ (RESERVED|DATA )
```

*where:***vname**

is the name of the virtual screen.

vline

is the line number of the virtual screen. A value of zero (0) indicates that the cursor is below the current bottom.

vcol

is the column number of the virtual screen.

RESERVED|DATA

is the area where the cursor is set.

DISPLAY

indicates the characteristics of the physical screen.

Responses:

```
DISPLAY lines cols devtype addrtype ducs color exthi pss pssets
```

*where:***lines**

is the number of lines on the physical screen.

cols

is the number of columns on the physical screen.

devtype

is the type of display terminal:

ANR - such as 3270 to 3277 type displays.

NDS - such as 3278, 3279, 3290 type displays.

addrtype

is either 12BIT type address or 14BIT type address.

dbcs

is DBCS if Double-Byte Character Set (DBCS) strings are supported, or NODBCS if DBCS strings are not supported.

color

is COLOR, if color is available, or NOCOLOR.

exthi

is EXTHI, if extended highlighting is available, or NOEXTHI.

pss

is PSS, if Programmed Symbol sets are available, or NOPSS.

pssets

is the list of Program Symbol sets (PSSET) that are currently loaded. Each character represents a PSSET. The PSSET value of 0 is always displayed.

FULLREAD

indicates whether or not 3270 null characters are recognized in the middle of the physical screen.

Response:

FULLREAD ON

or

FULLREAD OFF

where:

ON

indicates that null characters are recognized in the middle of lines, making it easier for you to enter tabular or pictorial data.

OFF

inhibits transmission of nulls from the terminal.

FULLSCREEN

indicates the status of full-screen CMS.

Response:

FULLSCREEN ON

or

FULLSCREEN OFF

or

FULLSCREEN SUSPEND

where:

ON

indicates that full-screen CMS is active.

OFF

indicates that full-screen CMS is inactive.

SUSPEND

indicates that full-screen CMS is suspended.

HIDE wname

displays the name of the window, the name of the virtual screen to which the window is connected, and the location (line and column) of the window on the virtual screen.

Responses:

WINDOW wname ON vname line column

where:

wname

is the name of the window.

QUERY

vname

is the name of the virtual screen to which the window is connected.

line

is the line number in the virtual screen where the window is connected.

column

is the column number in the virtual screen where the window is connected.

HIDE *

displays the name of the window, the name of the virtual screen to which the window is connected, and the location (line and column) of the window on the virtual screen for all hidden windows. This is the default.

Responses:

The response is the same as for QUERY HIDE *wname*; one line is displayed for each hidden window.

KEY

displays the last key pressed that caused an attention interrupt.

Responses:

KEY = keypressed

where:

keypressed

is the attention key that was pressed, such as ENTER, PAKEY *n*, PFKEY *n*, CLEAR, or UNKNOWN.

LINEND

indicates whether or not the logical line end character is activated and the character defined as the logical line end.

Response:

LINEND ON char

or

LINEND OFF char

where:

ON

indicates that the logical line end character is activated.

OFF

indicates that the logical line end character is not activated.

char

is the logical line end character.

LOCATION *wname*

indicates whether or not the location indicator is displayed in the specified window when the data in the virtual screen exceeds the size of the window.

Response:

LOCATION *wname* ON

or

LOCATION *wname* OFF

where:

wname

is the name of the specified window.

ON

the location indicator is displayed when there is data to be viewed outside the window.

OFF

the location indicator is not displayed.

LOGFILE vname

displays whether or not a log file is updated with data written to the virtual screen.

Response:

LOGFILE vname ON fn ft fm

or

LOGFILE vname OFF fn ft fm

where:

ON

indicates that a log file is updated with data written to the virtual screen.

OFF

indicates that a log file is not updated for the virtual screen.

vname

is the name of the virtual screen.

fn

is the file name of the log file.

ft

is the file type of the log file.

fm

is the file mode of the log file.

NONDISP

specifies the character that is set to be displayed in place of nondisplayable characters.

Response:

NONDISP char

where:

char

is the character that is displayed in place of nondisplayable characters.

REMOTE

displays the manner in which data transmission is handled.

Response:

REMOTE ON

or

REMOTE OFF

where:

ON

specifies that data is compressed by removing nulls and combining data when five or more of the same characters occur consecutively in a data stream.

OFF

specifies that the data stream is not compressed. Data is transmitted with no minimization.

RESERVED *wname*

displays the maximum number of reserved lines on the top and bottom of the specified window.

Response:

```
RESERVED wname rtop rbot
                *   *
```

where:

wname

is the name of the window.

rtop

is the maximum number of top reserved lines that may be displayed.

rbot

is the maximum number of bottom reserved lines that may be displayed.

ROUTE *msgclass*

displays the name of the virtual screen, alarm status, and notification information for the specified message class when SET FULLSCREEN is ON or SUSPEND. The following classes of output may be specified:

```
CMS
CP
MESSAGE
WARNING
SCIF
NETWORK
```

Response:

```
msgclass TO vname (alarm notify
```

where:

msgclass

is the message class which is directed to the virtual screen.

vname

is the virtual screen receiving the output.

alarm

is ALARM if the alarm is sounded when a message is received and NOALARM if the alarm does not sound when a message is received.

notify

is NOTIFY if the message class name is displayed in the status area when you receive a message and is NONOTIFY if the message class name is not displayed in the status area when you receive a message.

ROUTE *

displays the routing of all message classes. This is the default.

Response:

The response is the same as for QUERY ROUTE msgclass; one line is displayed for each message class.

SHOW *wname*

displays the name of the window, the name of the virtual screen that the window is showing, and the location (line and column) of the window on the virtual screen.

Note: If a variable size window is showing a virtual screen that does not contain any scrollable data lines or if a variable size window is clear, the window is not displayed on the physical screen. It does appear in the response.

Responses:

WINDOW *wname* ON *vname* *line* *column*

where:

wname

is the name of the window.

vname

is the name of the virtual screen to which the window is connected.

line

is the line number in the virtual screen where the window is connected.

column

is the column number in the virtual screen where the window is connected.

SHOW *

displays the name of the window, the name of the virtual screen to which the window is connected, and the location (line and column) of the window on the virtual screen for all windows in the order in which they are being displayed on the physical screen. This is the default.

Responses:

The response is the same as for QUERY SHOW *wname*; one line is displayed for each visible window.

TEXT

displays the status of TEXT character code conversion.

Response:

TEXT ON

or

TEXT OFF

where:

ON

converts TEXT characters for windows.

OFF

does not convert TEXT characters.

QUERY

VSCREEN *vname*

displays the name of the virtual screen, the number of lines and columns, and the number of top and bottom reserved lines for the specified virtual screen.

Responses:

```
VSCREEN vname lines cols rtop rbot
```

where:

vname

is the name of the virtual screen.

lines

is the number of lines in the virtual screen.

cols

is the number of columns in the virtual screen.

rtop

is the number of lines in the top reserved area.

rbot

is the number of lines in the bottom reserved area.

VSCREEN *

displays the name of the virtual screen, the number of lines and number of columns, and the number of top and bottom reserved lines for all virtual screens. This is the default.

Responses:

The response is the same as for **QUERY VSCREEN *vname***; one line is displayed for each virtual screen.

VSCREEN *vname* ALL

VSCREEN *

displays the attributes and extended attributes for a virtual screen or all virtual screens in addition to the same information as **QUERY VSCREEN**.

Response:

```
VSCREEN vname lines cols rtop rbot (high protect color exthi psset type system
```

where:

vname

is the name of the virtual screen.

high

is the intensity attribute of the data in the virtual screen. It may be **HIGH** or **NOHIGH**.

protect

indicates whether or not the data in the virtual screen is protected. It may be **PROTECT** or **NOPROT** (**NOPROTECT**).

color

is the color of the virtual screen.

exthi

is the extended highlighting of the virtual screen.

psset

is the Programmed Symbol set of the virtual screen.

type

indicates that data is moved to the virtual screen when the virtual screen queue is processed (TYPE) or that the virtual screen is not updated (NOTYPE).

system

indicates whether the virtual screen is retained (SYSTEM) or deleted (USER) when a task abnormally ends (abend) or when the HX (halt execution) command is issued.

Note: When using the ALL operand, the response may be too long for one line. If so, it wraps to the next one.

WINDOW *wname*

displays the name, the number of lines and columns, and the location on the physical screen for a specific window.

Responses:

WINDOW *wname* lines cols psline pscol

where:

wname

is the name of the window.

lines

is the number of lines in the window.

cols

is the number of columns in the window.

psline

is the line on the physical screen where the window is placed.

pscol

is the column on the physical screen where the window is placed.

WINDOW =

displays the name, the number of lines and columns, and the location on the physical screen for the topmost window.

Responses:

The response is the same as for QUERY WINDOW *wname*; one line is displayed for the topmost window.

WINDOW *

displays the name, the number of lines and columns, and the location on the physical screen for all windows. This is the default.

Responses:

The response is the same as for QUERY WINDOW *wname*; one line is displayed for each window.

WINDOW *wname* ALL

WINDOW * ALL

displays the options for a window or all windows in addition to the same information as QUERY WINDOW.

Response:

WINDOW *wname* lines cols psline pscol (type border pop top system

where:

QUERY

type

is **VARIABLE** if the window is a variable size or **FIXED** if the window is a fixed size.

border

is **BORDER** if the border is set **ON**, or **NOBORDER** if the borders are **OFF**.

pop

indicates whether the window is displayed on top of all other windows (**POP**) or that there is no effect on the window's position in the ordered list of windows (**NOPOP**) when the virtual screen that the window is showing is updated.

top

indicates whether the window may qualify as the topmost window (**TOP**) or cannot qualify as the topmost window (**NOTOP**)

system

indicates whether the window is retained (**SYSTEM**) or deleted (**USER**) when a task abnormally ends (abend) or when the **HX** (halt execution) command is issued.

Note: When using the **ALL** operand, the response may be too long for one line. If so, it wraps to the next one.

WMPF nn

displays the definition of a WMPF key specified as *nn*.

Responses:

WMPF nn pseudonym keyword string

*where:***pseudonym**

is a 9-character representation that is displayed in the PF Key definition area at the bottom of the WM window.

keyword

indicates when the command associated with the PF key is executed in relation to other commands entered at the terminal. It may be **DELAYED**, **ECHO**, or **NOECHO**. A WMPF key set to **RETRIEVE** does not have a keyword associated with it.

string

is the command string associated with the PF Key.

An undefined WMPF key is displayed as

WMPF nn

WMPF *

displays the definitions for all WMPF keys. This is the default.

Responses: The response is the same as for **QUERY WMPF nn**; one line is displayed for each PF key.

Options

STACK

causes the results of the QUERY command to be placed in the program stack instead of being displayed at the terminal. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

If CMS passes the command to CP, then the response from CP is also put in the program stack. If CP precedes the QUERY command, CMS does not stack the results. The STACK option is valid only when issued from CMS. Error messages are displayed at the terminal and are not stacked.

FIFO

(first-in first-out) is the default option for STACK. FIFO causes the results of the QUERY command to be placed in the program stack instead of being displayed at the terminal. The information is stacked FIFO. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO

(last-in first-out) causes the results of the QUERY command to be placed in the program stack rather than being displayed at the terminal. The information is stacked LIFO. This option is equivalent to STACK LIFO.

Usage Notes

1. You may specify only one QUERY parameter at a time.
2. If the implied CP (IMPCP) function is in effect and you enter an invalid QUERY parameter, you may receive the message DMKCG045E - userid NOT LOGGED ON.
3. If an invalid QUERY parameter is specified from an exec and the implied CP (IMPCP) function is in effect, then the return code is -0003.
4. If the information that you query requires a response from CP and the response is longer than 8192 characters, nothing is stacked and you receive a return code of 88.
5. The language for QUERY command responses depends on the language used to enter the command and whether or not the command was translated. If the QUERY command is entered in American English (or AMENG, which is always available), CMS responds in American English. If the command is entered in the current national language, or if the translation of the command is the same as American English, the response is displayed (or stacked) in the current national language. The language of the response is especially important for language dependent execs. For more information, see the section "Using Translations" in the *VM/SP CMS User's Guide*.

Messages and Return Codes

DMSJNL109S	Virtual storage capacity exceeded [RC = 104]
DMSQRF525E	Invalid PFkey number [RC = 24]
DMSQRF921E	Virtual screen <i>vname</i> is not defined [RC = 28]
DMSQRG918E	No windows are defined [RC = 4]
DMSQRG918E	No virtual screens are defined [RC = 4]
DMSQRG921E	Window <i>wname</i> is not defined [RC = 28]
DMSQRG1082E	No window qualifies as the window on top [RC = 4]
DMSQRH916E	Window <i>wname</i> is not displayed [RC = 28]
DMSQRH916E	Window <i>wname</i> is not hidden [RC = 28]
DMSQRH917E	No windows are displayed [RC = 4]
DMSQRH917E	No windows are hidden [RC = 4]
DMSQRP065E	<i>option</i> option specified twice [RC = 24]

QUERY

DMSQRQ1239E You are not authorized to issue this request on behalf of *userid*
[RC = 76]
DMSQRR109E Virtual storage capacity exceeded [RC = 109] Directory *dirname*
not found or you are not authorized for it
DMSQRX070E Invalid parameter *parameter* [RC = 24]
DMSQRY003E Invalid option: *option* [RC = 24]
DMSQRY014E Invalid function *function* [RC = 24]
DMSQRY066E *option1* and *option2* are conflicting options [RC = 24]
DMSQRY621E Bad plist: *message* [RC = 24]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

REFRESH

Use the REFRESH command to update virtual screens and their associated windows and refresh the screen.

Format

REFresh	
----------------	--

Messages and Return Codes

DMSREF391E Unexpected operand(s): *operand* [RC=24]
 DMSREF622E Insufficient free storage [RC=104]
 DMSREF917E No windows are displayed [RC=4]
 DMSREF925E I/O error on screen [RC=100]
 DMSREF926E Command is only valid on a display terminal [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

RESTORE WINDOW

RESTORE WINDOW

Use the RESTORE WINDOW command to return a maximized or minimized window to its size and location prior to the maximize or minimize.

Format

REStore WInDow	$\left[\begin{array}{c} wname \\ \underline{=} \end{array} \right]$
-----------------------	--

Operands

wname

is the name of the window to be restored. An "=" indicates that the topmost window will be restored. "=" is the default.

Messages and Return Codes

DMSRES386E Missing operand(s) [RC=24]
DMSRES388E Invalid keyword: *keyword* [RC=24]
DMSRES391E Unexpected operand(s): *operand* [RC=24]
DMSRES921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

ROUTE

Use the ROUTE command to direct data of a particular message class to a virtual screen.

Format

ROUTE	<pre>msgclass TO vname [(options ... [])]</pre> <p>Options: [<u>ALARM</u>] [<u>NOTify</u>] [<u>NOALARM</u>] [<u>NONotify</u>]</p>
--------------	---

Operands*msgclass*

identifies the message class which is being directed to the virtual screen. The following classes of output may be specified:

- | | |
|----------------|---|
| CMS | directs the responses generated by the virtual machine. The responses include any CMS error messages and line mode I/O performed by the virtual machine. |
| CP | directs messages and responses generated by CP. |
| MESSAGE | directs messages sent by the MSG and MSGNOH (message-noheader) commands from other users. |
| WARNING | directs warning messages sent by the CP WARNING command from the system operator. |
| SCIF | directs any messages from a secondary user ID to a virtual screen. |
| NETWORK | directs RSCS file routing messages from the RSCS virtual machine. All other RSCS messages are handled as regular messages. See Usage Note 4 for a list of the messages classified as NETWORK. |
| * | directs all classes of information to a specified virtual screen. |

vname

specifies which virtual screen receives the output.

Options

The options apply to the MESSAGE, WARNING, SCIF, and NETWORK message classes. Although you can specify the options for CP and CMS, they are ignored.

ALArm

sounds the alarm when a message is received.

NOAlarm

does not sound the alarm. NOALARM is the default.

NOTify

displays the message class name in the status area when you receive a message with a message class of MESSAGE, WARNING, SCIF, or NETWORK.

ROUTE

NONotify

will not display the virtual screen name in the status area when you receive a message with a message class of MESSAGE, WARNING, SCIF, or NETWORK. NONOTIFY is the default.

Usage Notes

1. When SET FULLSCREEN is ON, the various message classes are routed to virtual screens as follows:

Message Class	Virtual Screen	Options
CMS	CMS	NOALARM NONOTIFY
CP	CMS	NOALARM NONOTIFY
MESSAGE	MESSAGE	ALARM NOTIFY
WARNING	WARNING	ALARM NOTIFY
SCIF	MESSAGE	NOALARM NONOTIFY
NETWORK	NETWORK	NOALARM NOTIFY

Commands entered in the CMS virtual screen are always echoed in the CMS virtual screen regardless of the routing of the CMS message class.

2. The CP and CMS message classes always have options of NOALARM and NONOTIFY.
3. When you specify the message class as *, the options are recognized for MESSAGE, WARNING, SCIF, and NETWORK. CP and CMS always have the NOALARM and NONOTIFY options.
4. The following messages from the Remote Spooling Communications Subsystem (RSCS) are classified as NETWORK messages:

```
101I FILE spoolid ENQUEUED ON LINK linkid
104I FILE (orgid) {SPOOLED|TRANSFERRED} TO userid1 ORG
      nodeid(userid2) mm/dd/yy hh:mm:ss: zzz
147I SENT FILE spoolid(orgid) ON LINK linkid TO nodeid(userid2)
170I FROM nodeid: message-text
```

All other RSCS messages are handled as regular messages.

The first three characters of the identifier of an RSCS message are DMT. The next three characters denote the module origin of the message. For more information on messages from RSCS, see *Remote Spooling Communications Subsystem Program Reference and Operations Manual* or *Remote Spooling Communications Subsystem Networking Version 2 Operation and Use*.

Messages and Return Codes

DMSROU386E Missing operand(s) [RC=24]
 DMSROU388E Invalid keyword: *keyword* [RC=24]
 DMSROU391E Unexpected operand(s): *operand* [RC=24]
 DMSROU394E Invalid option: *option* [RC=24]
 DMSROU921E Virtual screen *vname* is not defined [RC=28]
 DMSROU926E Command is only valid in CMS FULLSCREEN mode [RC=88]
 DMSROU928E Command is not valid for virtual screen *vname* [RC=12]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SCROLL

Use the SCROLL command to move a window to a new location on the virtual screen to which it is connected.

Format

SCROLL	$\left[\begin{array}{l} \text{Backward} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ * \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \\ \text{Bottom} \left[\begin{array}{l} \textit{wname} \\ = \end{array} \right] \\ \text{Down} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ * \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \\ \text{Forward} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ * \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \\ \text{Left} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \\ \text{Next} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ * \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \\ \text{Right} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \\ \text{Top} \left[\begin{array}{l} \textit{wname} \\ = \end{array} \right] \\ \text{Up} \left[\begin{array}{l} \textit{wname} \left[\begin{array}{l} n \\ * \\ \underline{1} \end{array} \right] \\ = \end{array} \right] \end{array} \right]$
---------------	--

Operands

wname

is the name of the window being scrolled. An "=" indicates that the topmost window is scrolled. If this operand is not specified, "=" is assumed.

Backward

moves the window toward the top of the virtual screen so that the first data line displayed in the window becomes the last data line displayed. If you specify a number for *n*, the window is scrolled backward "*n*" displays. The * means scroll to the top. The default is 1 display.

Bottom

moves the window to the last line (bottom) of the virtual screen. The first data line of the window displays the last line of the virtual screen.

Down

moves the window toward the bottom of the virtual screen the number of data lines specified. The default is 1 line. DOWN is the same as NEXT. The * means scroll to the bottom.

Forward

moves the window toward the bottom of the virtual screen so that the last data line displayed in the window becomes the first data line displayed. If you specify a number for *n*, the window is scrolled forward “*n*” displays. The default is 1. The * means scroll to the bottom.

Left

moves the window toward the left edge of the virtual screen the number of columns specified. The default for *n* is 1. If *n* is specified as 0, the window is moved to column 1.

Next

moves the window toward the bottom of the virtual screen the number of lines specified. The default is 1 line. NEXT is the same as DOWN. The * means scroll to the bottom.

Right

moves the window toward the right edge of the virtual screen the number of columns specified. The default for *n* is 1. If *n* is specified as 0, the window is moved to column 1.

Top

moves the window to the first line (top) of the virtual screen. The first data line of the window displays the first line of the virtual screen.

Up

moves the window toward the top of the virtual screen the number of data lines specified. The default is 1 line. The * means scroll to the top.

Usage Notes

1. The SCROLL command is valid only when the window is connected to a virtual screen (see the SHOW WINDOW and HIDE WINDOW commands).
2. When you scroll forward or backward, only the data area of the screen moves; the reserved lines (PF key definition area and title line) are static. When you scroll left or right, the reserved lines of the screen are scrolled along with the data area of the screen.
3. If the window has been cleared (see the CLEAR WINDOW command), the following rules apply:
 - If you SCROLL FORWARD or TOP, you scroll to the top of the virtual screen
 - If you SCROLL BACKWARD or BOTTOM, you scroll to the bottom of the virtual screen
 - SCROLLing UP or DOWN has no effect.
4. If the window is positioned in the middle of the virtual screen and you scroll backward specifying an “*n*,” which would result in scrolling past the virtual screen top, the window is repositioned at the virtual screen top and stops. If the

SCROLL

window is positioned at the virtual screen top and you scroll backward, the window is repositioned at the virtual screen bottom and stops.

5. If the window is positioned in the middle of the virtual screen and you scroll forward specifying an "n," which results in scrolling past the bottom, the window is repositioned at the virtual screen bottom, and stops. If the window is positioned at the virtual screen bottom and you scroll forward, the window is cleared. A subsequent scroll forward positions the window at the virtual screen top. If the window being scrolled is variable size and it is positioned at the bottom of the virtual screen, the window is not displayed at the next refresh when it is scrolled forward (see the CLEAR WINDOW command).
6. When you receive the status area message "Scroll forward for more information in vscreen *wname*" (in full-screen CMS) and there are multiple windows showing the specified virtual screen, it is recommended that you scroll forward the window closest to the top of the ordered list of windows. This enables data that is in the virtual screen queue to be written to the virtual screen and displayed.

Messages and Return Codes

DMSSCL386E	Missing operand(s) [RC=24]
DMSSCL388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSSCL389E	Invalid operand: <i>operand</i> [RC=24]
DMSSCL391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSSCL921E	Window <i>wname</i> is not defined [RC=28]
DMSSCL929E	Window <i>wname</i> is not connected to a virtual screen [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

Return Codes

- | | |
|---|---|
| 0 | Normal |
| 1 | Vscreen top or bottom reached, or window is cleared |
| 3 | Scroll amount truncated |

SET

Use the SET command to establish, turn off, or reset a particular function in your CMS windowing or virtual screen environment. Only one function may be specified per SET command. SET cannot be issued without an option.

The full-screen CMS options available with SET are summarized below. A complete description of each option follows.

APL	LOGFILE
BORDER	NONDISP
CHARMODE	REMOTE
CMSPF	RESERVED
FULLREAD	TEXT
FULLSCREEN	VSCREEN
LINEND	WINDOW
LOCATION	WMPF

Usage Notes

1. If you issue the SET command specifying an invalid function and the implied CP function is in effect, you may receive message DMKCFC003E Invalid option - *option*
2. If an invalid SET command function is specified from an exec and the implied CP function is in effect, then the return code is -0003.
3. To determine or verify the setting of most functions, use the QUERY command.

Messages and Return Codes

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET APL

Use the APL option to activate character code conversion for APL characters for the System Product Editor and CMS.

Format

SET	APL { ON OFF }
------------	--

Operands

ON

activates character code conversion for APL characters. Before using APL keys, issue SET APL ON to ensure proper character code conversion.

OFF

specifies that no character code conversion is performed for APL characters and keys.

Initial Setting

APL OFF

Usage Notes

1. The APL setting is valid only when performing full-screen I/O (for example, in XEDIT or in CMS with SET FULLSCREEN ON). If you are in CP or using a line-mode terminal, SET APL has no effect.

If you are in CP, you can issue the TERMINAL APL ON command to have CP convert APL character codes.

2. Because the APL character code conversion is costly, it is recommended that you issue SET APL OFF when you stop using the special APL keys.
3. When SET APL ON is specified, TEXT is set OFF.
4. Changing the APL setting for CMS also changes the APL setting for the System Product Editor, and vice versa.

Messages and Return Codes

DMSSEF109S	Virtual storage capacity exceeded [RC = 104]
DMSSEF524W	NONDISP character reset to ".
DMSSEF926E	Command is only valid on a display terminal [RC = 88]
DMSSET109S	Virtual storage capacity exceeded [RC = 104]
DMSWIR329W	Warning: APL/TEXT option not in effect

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET BORDER

Use the **BORDER** option to define borders around windows. Borders visually separate information displayed in different windows. Also, border corners can be used to enter Border Commands (see “Border Commands” on page 778 for more information).

Format

SET	BORDER <i>wname</i> { ON OFF } [([optionA] [optionB] [])] OptionA: [TOP <i>char</i>] [BOTTOM <i>char</i>] [LEFT <i>char</i>] [RIGHT <i>char</i>] [ALL <i>char</i>] OptionB: [High] [<i>color</i>] [<i>exthi</i>] [<i>psset</i>] [NOHigh]
------------	--

Operands

wname

is the name of the window.

ON

indicates that the borders of the window will be displayed (if they fit on the physical screen). If no options are specified, the currently defined border characters and attributes are used.

OFF

indicates that the borders will not be displayed.

OptionA

allows you to specify new characters for window borders. One or more of the following options may be specified:

TOP *char*

where *char* is the character that is displayed in the top border.

BOTTOM *char*

where *char* is the character that is displayed in the bottom border.

LEFT *char*

where *char* is the character that is displayed in the left-hand border.

RIGHT *char*

where *char* is the character that is displayed in the right-hand border.

ALL *char*

where *char* is the character that is displayed in all borders.

SET BORDER

OptionB

indicates how the borders should be displayed. The following can be specified:

High

borders are high intensity.

NOHigh

borders are normal intensity.

color

is the border color. It may be Default, Blue, Red, Pink, Green, Turquoise, Yellow, or White.

exthi

is the border extended highlighting. It may be None, REVvideo, BLInk, or Underline.

psset

is the Programmed Symbol Set (PSset) used to display the borders (PS0, PS1, PSA, PSB, PSC, PSD, PSE, or PSF). The PSset must be loaded in the display to be used. If not, the default PSset is used.

Initial Setting

ON or OFF

Usage Notes

1. To override the default border characteristics (see DEFINE WINDOW for the defaults) or to display a particular edge of the border you must use option A. Only those edges specified in option A are set "ON." All other edges are set "OFF."

For example, if you issue the command:

```
set border message on (top *
```

only the top border is displayed and it is all asterisks (*). The bottom, left, and right borders are *not* displayed.

The settings specified remain in effect for the duration of the session or until you change them.

2. Window borders are built outside the area defined for the window. Therefore, due to the size and position of the window, it is possible that any or all of the borders may not fit on the physical screen.

Note: Left and right border characters take up two columns on the physical screen. (One column is for a start field and another column is for the border character.) Top and bottom borders take only one line. Thus, on a 24 x 80 physical screen, a window must not start before column 3 and must not extend past column 78 for the left and right borders to be displayed. To display top and bottom borders, the window must not start before line 2 or extend past line 23.

3. The corner characters of the border (identified by plus (+) signs) are available for entering single character windowing or scrolling commands (see the Border Commands section of this book). All other border characters are protected.
4. Location information for the number of lines or columns, or both, is displayed using the color, highlight, and program symbol set defined for the window border.

Messages and Return Codes

DMSSEF921E Window *wname* is not defined [RC = 28]
DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET CHARMODE

Use the CHARMODE option to specify whether character attributes or field attributes should be used when displaying virtual screen data on the physical screen.

Format

SET	CHARMODE { ON OFF }
------------	---

Operands

ON

displays each character with its own attribute. Each displayed character can be given individual color, extended highlighting, and PSset.

OFF

displays each character in a field with the attributes of the field. In this case, individual character attributes are ignored.

Initial Setting

CHARMODE OFF

Usage Notes

1. The structure of virtual screens allows you to give different attributes to each character. For example, adjacent characters can be displayed with different colors. See the WRITE VSCREEN command for a description of how to specify character attributes when writing data.
2. To use character attributes, the display device must support character attributes. If the device does not support them (for example, the terminal is a 3277), then field attributes are used regardless of the CHARMODE setting. For more information, refer to the *IBM 3270 Information Display System Data Stream Programmer's Reference*, GA23-0059.
3. SET CHARMODE must be ON to update COLOR, EXTHI, and PSS in the WRITE VSCREEN command. If SET CHARMODE is OFF they are ignored. Switching from SET CHARMODE ON to OFF may produce some undesirable results, such as a field having attributes that you intended only for a character.
4. For color and extended highlighting in a DBCS string, the first byte of a double-byte character determines the attributes for both bytes. You cannot specify character attributes for PSS in the WRITE VSCREEN command within a DBCS string.

Messages and Return Codes

DMSSEF109S	Virtual storage capacity exceeded [RC=104]
DMSSEF926E	Command is only valid on a display terminal [RC=88]
DMSSET109S	Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

	Reason	Page
	Errors in command syntax	811

SET CMSPF

Use the CMSPF option to set a CMSPF key to a specific command. The CMSPF keys are used when SET FULLSCREEN is ON.

Format

SET	CMSPF <i>nn</i> [{ <i>pseudonym</i> } [<i>keyword</i> DELAYED] <i>string</i>]
------------	---

Operands*nn*

is a number from 1 to 24 indicating which PF key is being set.

pseudonym

is a 9-character representation of the PF key definition. The pseudonym is displayed in the PF key definition area at the bottom of the CMS window. The pseudonym may be up to nine characters in length.

NOWRITE

suppresses overwriting of the PF key pseudonym when you set a CMSPF key.

keyword

indicates when the command associated with the PF key is executed in relation to other commands entered at the terminal. The *keyword* may be any of the following:

DELAYED

delays the execution of the command string. When the key is pressed, the command is displayed in the input area and is not executed until you press the ENTER key. If anything is currently in the input area, it is overlaid and no commands entered on the physical screen are processed. DELAYED is the default setting if no keyword is specified on the SET CMSPF command.

ECHO

executes the command immediately when the program function key is pressed. The key definition is echoed on the CMS virtual screen.

NOECHO

executes the command immediately when the program function key is pressed. The key definition is not echoed on the CMS virtual screen.

Note: When a CMSPF key is set to RETRIEVE the keyword is ignored.

string

is the command(s) to be executed when the key is pressed.

Initial Setting

CMSPF 01 Help	ECHO	HELP
CMSPF 02 Pop_Msg	NOECHO	POP WINDOW MESSAGE *
CMSPF 03 Quit	NOECHO	SET FULLSCREEN SUSPEND
CMSPF 04 Clear_Top	NOECHO	#WM CLEAR WINDOW =
CMSPF 05 Filelist	ECHO	EXEC FILELIST

CMSPF 06 Retrieve		RETRIEVE
CMSPF 07 Backward	NOECHO	#WM SCROLL BACKWARD CMS 1
CMSPF 08 Forward	NOECHO	#WM SCROLL FORWARD CMS 1
CMSPF 09 Rdrlist	ECHO	EXEC RDRLIST
CMSPF 10 Left	NOECHO	#WM SCROLL LEFT CMS 10
CMSPF 11 Right	NOECHO	#WM SCROLL RIGHT CMS 10
CMSPF 12 Cmdline	NOECHO	CURSOR VSCREEN CMS -2 8 (RESERVED)

Note: On a terminal equipped with 24 PF keys, PF keys 13 through 24 have the same values as PF keys 1 through 12, respectively.

Usage Notes

1. You can assign a sequence of commands to a single PF key by:
 - a. Setting off the LINEND character
 - b. Setting the PF key to the commands separated by the LINEND character
 - c. Setting the LINEND character ON before using the PF key

You cannot assign a sequence of #WM commands to a CMSPF key. The SET CMSPF command will accept the sequence, but the commands will not execute.

2. To cancel a PF key definition, enter:


```
set cmspf nn
```

 substituting the number of the PF key for nn.
3. When you press a PA key or a CMSPF key in the CMS window, any input on the screen that has not been processed is discarded except input that is typed on the command line. If the key that was pressed does not update the command line, then input on the command line is rewritten. The next time you press ENTER it is executed.
4. The RETRIEVE function saves previously entered commands in a buffer that is 256 characters long. When you enter full-screen CMS, the buffer contains an asterisk (comment), and commands are added to the buffer as they are entered until it is full. As you continue to enter commands, the oldest commands are deleted and the most current commands are added.

Pressing the PF key assigned to RETRIEVE displays the next command in the buffer on the command line. Each time you press the key, the previously entered command is displayed until the oldest one is reached. Then, RETRIEVE returns the most current command. Once the command is on the command line, press ENTER to execute it. You may also modify the command, then press ENTER to execute the new command.
5. The NOWRITE option is particularly useful when you have changed the bottom reserved area in the CMS virtual screen and you do not want the area overwritten when you set a CMSPF key. However, when you enter the full-screen CMS environment for the first time, the CMSPF key definitions are overwritten in the bottom reserved area of the CMS virtual machine.

SET CMSPF

Messages and Return Codes

DMSSEF525E Invalid PFkey number [RC=24]
DMSSEF926E Command is only valid in CMS FULLSCREEN mode [RC=88]
DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET FULLREAD

Use the FULLREAD option to allow 3270 null characters to be recognized in the middle of the physical screen by CMS and the System Product Editor.

Format

SET	FULLREAD { ON OFF }
-----	------------------------

Operands

ON

enables nulls to be recognized in the middle of lines, making it easier for you to enter tabular or pictorial data.

OFF

inhibits transmission of nulls from the terminal.

Initial Setting

OFF

Usage Notes

1. When FULLREAD ON is issued, nulls at the end of screen lines that are part of a logical line that occupies more than one physical screen line are dropped. This allows you to delete characters in a screen line and still have the line reconstructed flush together even though multi-line 327X lines do not “wrap” when the character delete key (or the insert mode key) is used.
2. FULLREAD ON increases communication to the CPU, which generally results in increased response time.
3. Setting FULLREAD ON will prevent you from losing any screen changes when you press a PA key and a message is displayed on a cleared screen.
4. A certain terminal configuration, which imposes several restrictions on your session, occurs when going through a VM/Passthru Facility (5749-RC1) (PVM) 327X Emulator link to another VM system. These PVM links can be identified by an S to the immediate left of the node ID in the PVM selection screen. The following is a list of these restrictions:
 - a. The SET FULLREAD ON command may not be used.
 - b. All PA keys (except for the CP defined TERMINAL BRKKEY) are non-functional.
5. Changing the FULLREAD setting for CMS also changes the FULLREAD setting for the System Product Editor.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
DMSSEF926E Command is only valid on a display terminal [RC = 88]
DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET FULLSCREEN

Use the FULLSCREEN option to use full-screen CMS.

Format

SET	FULLSCREEn { ON OFF SUSPEND RESUME } [(options ... [])]
	<u>Options:</u> [CLear] [NOCLear]

Operands

ON

initializes full-screen CMS. SET FULLSCREEN ON defines the default virtual screens and windows for CMS. Output that is normally displayed by CP is trapped by CMS (by the IUCV Message All System Service) so that messages and VM output are displayed in windows.

OFF

returns CMS to line-mode operation.

SUSPEND

specifies that CMS should temporarily return to line-mode operation. CMS discontinues trapping I/O (by severing the *MSGALL connection) so that CP displays messages and VM output. This option could be used by applications that perform their own full-screen management, such as those that use DIAGNOSE Code X'58'.

RESUME

returns a CMS session to the state that preceded a SET FULLSCREEN SUSPEND command.

Options

CLear

clears the screen and enters full-screen CMS when used with ON or RESUME. The screen is not placed in a "MORE..." status. CLEAR is the default.

NOCLear

does not clear the screen when used with ON or RESUME. The screen is placed in a "MORE..." status, and any messages remain on the screen until the CLEAR key is pressed. The NOCLEAR option is particularly useful when SET FULLSCREEN ON or SET FULLSCREEN RESUME is issued from an exec or program that displays messages.

Initial Setting

FULLSCREEN OFF

Things You Should Know About Full-Screen CMS

1. When you issue SET FULLSCREEN ON, CMS is placed in full-screen mode with the default and/or user-defined windows and features (for example: CMSPF Keys, Command Line, Status Area, etc.) being displayed.
2. When entering full-screen CMS:
 - All default virtual screens that you have not defined are defined, such as a virtual screen for CMS and CP output, and virtual screens for messages, network messages, warnings from the operator, and status information
 - All reserved areas are written for the default virtual screens
 - All default windows that you have not defined are defined
 - Default windows are connected to appropriate virtual screens
 - CMSPF key definitions are established
 - A connection to the IUCV Message All System Service is established and various classes of output are routed to virtual screens
 - Logging is started for the MESSAGE and WARNING virtual screens
 - The cursor is set in the CMS virtual screen
 - The CP TERMINAL BRKKEY NONE command is issued.
3. When returning to full-screen CMS after it has been suspended:
 - The CMS window is shown
 - The STATUS window is displayed
 - A connection is reestablished to the IUCV Message All System Service
 - Window and virtual screen definitions, logging, message routing, and the CP TERMINAL BRKKEY setting are not affected.
4. You must SET FULLSCREEN to OFF or SUSPEND to allow an APPC/VM application to connect to a private resource in your virtual machine. If FULLSCREEN is ON, CMS rejects any private resource connection requests.

Your virtual machine may be logged on automatically if it processes private resource connection requests. If your virtual machine processes private resource connection requests, put the statement SET FULLSCREEN OFF or SET FULLSCREEN SUSPEND in your PROFILE EXEC to make sure CMS does not reject any private resource connection requests.
5. When lines are written to a virtual screen sequentially, such as in the CMS virtual screen, lines are added to the virtual screen starting at the virtual screen top. Once the virtual screen is full and you scroll forward, the oldest lines that have been scrolled are deleted, new lines are appended at the bottom, and the lines are renumbered. (Lines that have not been scrolled are not deleted.) Because the lines are renumbered, the scroll location information may appear to remain the same as you scroll forward.

When the virtual screen is full and new information is waiting to be added, scrolling the virtual screen forward or entering one of the following commands: CLEAR WINDOW, CLEAR VSCREEN, SHOW WINDOW, or HIDE WINDOW, allows the virtual screen to be updated. That is, the oldest information that has been scrolled is deleted off the top so that the newest information can be added at the bottom. This updating process occurs even if the window connected to the virtual screen is hidden or overlaid by other windows.

6. When you receive the status area message “Scroll forward for more information in vscreen *vname*” and there are multiple windows showing the specified virtual screen, it is recommended that you scroll forward the window closest to the top of the ordered list of windows. This enables data that is in the virtual screen queue to be written to the virtual screen and displayed.
7. When you SET FULLSCREEN OFF, all information that has not been updated to a virtual screen prior to execution of the command will be typed out in line mode. Any default windows and virtual screens defined by full-screen CMS will be deleted.

In addition, the CP TERMINAL BRKKEY remains as NONE. To reset it to PA1 (the default setting), use the CP TERMINAL command.

8. Commands can only be entered in the CMS virtual screen and the WM window. Commands entered in the CMS virtual screen are always echoed in the CMS virtual screen regardless of the routing of the CMS message class.
9. You must always have a window showing the CMS virtual screen when using full-screen CMS. If you hide all the windows showing the CMS virtual screen, the CMS window is automatically shown at the top of the CMS virtual screen. The CMS window is on top of all other windows, including the STATUS window. You should then issue the POP WINDOW STATUS command.
10. The WM window is automatically displayed in full-screen CMS when no windows are showing the active virtual screen. For example, you may maximize a window so that it fills the entire screen and covers all other windows. You may not be able to enter commands in the window because it is protected. In such cases, the WM window is automatically displayed, and the WMPF keys and command line are available to manipulate the window.

In addition, the WM window is automatically displayed in full-screen CMS when you run an application that uses the CONSOLE macro to perform I/O and

- The CMS virtual screen is updated, or
- Any virtual screen (other than CMS) is updated and a pop-type window is showing it.

The WMPF keys and a command line are available. Issuing the DROP WINDOW WM command (default WMPF 3) returns you to the application.

System Defaults for Full-Screen CMS

1. The default windows for full-screen CMS are:

Table 21. Default Windows					
Wname	Lines	Cols	Psline	Pscol	Options
STATUS	1	Pscr	-1	1	Fixed Noborder Nopop Notop
CMS	Pscr	Pscr	1	1	Fixed Border Nopop Top
NETWORK	8	71	-12	7	Variable Border Nopop Top
WARNING	6	71	3	3	Variable Border Pop Top
MESSAGE	8	71	11	3	Variable Border Pop Top
WM	5	Pscr	-1	1	Fixed Border Nopop Notop
CMSOUT	8	75	9	3	Variable Border Pop Top

Pscr

Size of the physical screen

Psline

The line on the physical screen where the upper (when psline is positive) or lower (when psline is negative) edge of the window is placed.

Pscol

The column on the physical screen where the upper left corner of the window is placed.

Variable

indicates that the number of lines in the window may vary depending on the amount of scrollable data displayed.

Fixed

indicates that the number of lines in the window is always constant.

Border

indicates that the borders are displayed when possible. For the CMS window, the borders are on but you cannot see them because the window is the size of the physical screen.

Noborder

indicates that borders are not displayed.

Pop

specifies that the window is displayed on top of all other windows when the virtual screen that the window is showing is updated.

Nopop

specifies that there is no effect on the window's position in the ordered list of windows when the virtual screen that the window is showing is updated.

Top

specifies that the window may qualify as the topmost window.

Notop

specifies that the window cannot qualify as the topmost window

Note: Although the WM window is a default window, it is not defined when you enter full-screen CMS. The WM window is defined when you issue the command POP WINDOW WM, press the PA1 key, or when the WM window is automatically displayed.

All default windows are SYSTEM windows, which means the window is retained when the system abnormally terminates (abend) or when the HX (halt execution) command is issued.

- The default virtual screens for full-screen CMS are:

Vname	Lines	Cols	Rtop	Rbot	Dcolor	Protected?*
WM	1	Pscr	0	5	White	No
STATUS	1	Pscr	0	0	White	Yes
NETWORK	16	70	2	0	Blue	Yes
WARNING	4	70	2	0	Red	Yes
MESSAGE	20	70	2	0	White	Yes
CMS	120	Pscr	2	5	Green	No

Pscr

Physical screen size. For terminals with a screen size of 80 columns or less, the CMS virtual screen contains 81 columns. For terminals with a screen size greater than 80 columns, the CMS virtual screen contains the same number of columns as the physical screen. The STATUS and WM virtual screens always contain the same number of columns as the physical screen.

Rtop

Top reserved lines

Rbot

Bottom reserved lines

Dcolor

Data color

*

If protected, you cannot type into the window(s) connected to the virtual screen.

SET FULLSCREEN

Note: Although the WM virtual screen is a default virtual screen, it is not defined when you enter full-screen CMS. The WM virtual screen is defined when you issue the command POP WINDOW WM, press the PA1 key, or when the WM window is automatically displayed.

All default virtual screens are TYPE and SYSTEM virtual screens. TYPE means data is moved to the virtual screen when the virtual screen is updated. SYSTEM means the virtual screen is retained when the system abnormally terminates (abend) or when the HX (halt execution) command is issued.

3. Default windows are connected to default virtual screens in the following manner:

Window	Virtual Screen	Description
CMS	CMS	Displays CMS and CP output
CMSOUT	CMS	Displays CMS and CP output while in XEDIT
MESSAGE	MESSAGE	Displays user messages and SCIF messages
NETWORK	NETWORK	Displays network messages
STATUS	STATUS	Displays status messages
WARNING	WARNING	Displays warnings
WM	WM	Provides the capability to enter windowing commands

4. When SET FULLSCREEN is ON, the various message classes are routed to virtual screens as follows:

Message Class	Virtual Screen	Options
CMS	CMS	NOALARM NONOTIFY
CP	CMS	NOALARM NONOTIFY
MESSAGE	MESSAGE	ALARM NOTIFY
WARNING	WARNING	ALARM NOTIFY
SCIF	MESSAGE	NOALARM NONOTIFY
NETWORK	NETWORK	NOALARM NOTIFY

ALARM

sounds the alarm when a message is received.

NOALARM

does not sound the alarm.

NOTIFY

displays the message class name in the status area when you receive a message.

NONOTIFY

will not display the virtual screen name in the status area when you receive a message.

See the ROUTE command for information on changing the default message routing.

Commands entered in the CMS virtual screen are always echoed in the CMS virtual screen regardless of the routing of the CMS message class.

5. Any messages or warnings received during your full-screen CMS session are displayed in windows and logged into files. Messages are logged into a file called MESSAGE LOGFILE, and warnings are logged into a file called WARNING LOGFILE.
6. Pressing the PA1 key while in full-screen CMS displays the WM window. The PA2 key and CLEAR key scroll the topmost window forward. See the SET CMSPF command for the default settings for the CMSPF keys.

In the WM window, the PA2 key and CLEAR key also scroll the topmost window forward. When there is no more data in the window to scroll, you automatically exit the WM environment.

Considerations for Applications in Full-Screen CMS

1. If an application performs full-screen management while in full-screen CMS and it does not use the CONSOLE macro, the output written to full-screen CMS is not displayed until the application completes. Prior to running the application, issue the SET FULLSCREEN SUSPEND command; when the application completes, issue the SET FULLSCREEN RESUME command.
2. When returning to full-screen CMS from an application that performs its own full-screen management (such as DIAGNOSE Code X'58'), your screen may contain mixed data. Press the CLEAR key to scroll forward and refresh the screen.

Alternatively, issue the SET FULLSCREEN SUSPEND command, run the application, and then issue SET FULLSCREEN RESUME when the application completes. For more information, see *VM System Facilities for Programming*.

3. SET FULLSCREEN SUSPEND and SET FULLSCREEN RESUME can be "nested." For example, suppose full-screen CMS is ON and an application called MYPROG issues SET FULLSCREEN SUSPEND and calls another application, TESTPROG. TESTPROG also issues SET FULLSCREEN SUSPEND. When TESTPROG completes, it issues SET FULLSCREEN RESUME and control returns to MYPROG. SET FULLSCREEN is still in the SUSPEND state and MYPROG continues to execute. Upon completion MYPROG issues SET FULLSCREEN RESUME, which returns FULLSCREEN to the ON state.

To preserve the nesting, do not issue ON or OFF between SUSPEND and RESUME. If an application issues SET FULLSCREEN ON or OFF, the nesting is cleared and FULLSCREEN status changes to ON or OFF. Use the QUERY FULLSCREEN command to determine the FULLSCREEN status.

SET FULLSCREEN

4. If full-screen CMS has never been set ON, and either SET FULLSCREEN OFF, SET FULLSCREEN SUSPEND or SET FULLSCREEN RESUME is issued, no action is taken.
5. The following messages are not trapped by the IUCV Message All System Service and are sent directly to the terminal:
 - Asynchronous CPCONIO, including PER/TRACE events
 - EMSGs *not* generated as part of a DIAGNOSE code X'08' operation instruction.
 - Accounting messages
6. The IUCV Message All System Service can stack up to 255 messages at any one time. If this limit is exceeded, any additional incoming messages are sent directly to the terminal.
7. When SET FULLSCREEN is ON, most CMS console output is not passed to CP. In addition, applications that use the IUCV Message System Service (*MSG) and SET VMCONIO to IUCV will not trap all CMS output when using full-screen CMS. Prior to using such applications, it is recommended to issue SET FULLSCREEN SUSPEND.
8. You must SET FULLSCREEN to OFF or SUSPEND to allow an APPC/VM application to connect to a private resource in your virtual machine. If FULLSCREEN is ON, CMS rejects any private resource connection requests.

Your virtual machine may be logged on automatically if it processes private resource connection requests. If your virtual machine processes private resource connection requests, put the statement SET FULLSCREEN OFF or SET FULLSCREEN SUSPEND in your PROFILE EXEC to make sure CMS does not reject any private resource connection requests.
9. When developing an application to be used with full-screen CMS, you may want to reset the CP TERMINAL BRKKEY to PA1 (it is set to NONE in full-screen CMS). Then, you can enter CP mode to debug the application.

Messages and Return Codes

DMSSEF109S	Virtual storage capacity exceeded [RC=104]
DMSSEF926E	Command is only valid on a display terminal [RC=88]
DMSSEF927E	The physical screen must contain at least 20 lines and 80 columns [RC=24]
DMSSET109S	Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

Special Command Used in the Full-Screen Environment

The #WM command is a special command that can only be used in the CMS virtual screen in full-screen CMS.

#WM

Use the #WM command to execute a command immediately from the CMS virtual screen.

Format

#WM	<i>wmcommand</i>
-----	------------------

Operands*wmcommand*

specifies the command that you want to execute. See Usage Note 2 for a list of commands you can specify with #WM.

Usage Notes

1. The pound sign (#) represents the default logical line end symbol for full-screen CMS. If you have redefined the line end symbol to another character (using the SET LINEND command), #WM is an invalid command; you must substitute your line end symbol for the pound sign to use the command.

The #WM command is independent of the CP logical line end symbol.

2. You can enter any of the following commands with #WM:

CLEAR WINDOW	QUERY BORDER	SCROLL
CP	QUERY HIDE	SET BORDER
DROP WINDOW	QUERY LOCATION	SET LOCATION
HIDE WINDOW	QUERY RESERVED	SET RESERVED
MAXIMIZE WINDOW	QUERY SHOW	SET WINDOW
MINIMIZE WINDOW	QUERY WINDOW	SET WMPF
POP WINDOW	QUERY WMPF	SHOW WINDOW
POSITION WINDOW	RESTORE WINDOW	SIZE WINDOW
PUT SCREEN		

Note that HELP is not a valid command with #WM; however, it *is* a valid command in the WM environment.

Messages and Return Codes for #WM

DMSWEN931E Invalid WM command: *command*

DMSWEN1125E *command* is not allowed as an immediate command

SET LINEND

Use the LINEND option to activate and/or define the logical line end for full-screen CMS.

Format

SET	LINEND { ON OFF } [<i>char</i>]
------------	---

Operands

ON

allows you to enter several commands on the same line, separated by the line end character.

OFF

specifies that the logical line end character is not recognized.

char

is the character to be used as a line end character. The default line end character is a pound sign (#).

Initial Setting

LINEND ON #

Usage Note

If you redefine the line end character to a symbol other than a pound sign (#), the #WM command is invalid. (Therefore, the default CMSPF keys that issue #WM commands do not function.) To use the #WM command you must substitute your line end symbol for the pound sign.

Messages and Return Codes

DMSSEF926E Command is only valid in CMS FULLSCREEN mode [RC=88]
 DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET LOCATION

Use the LOCATION option to display the location indicator in the window when the data in the virtual screen exceeds the size of the window.

Format

SET	LOCATION <i>wname</i> { ON OFF }
------------	--

Operands

wname

is the name of the window.

ON

displays the location indicator when there is data to be viewed outside of the window.

OFF

does not display the location indicator.

Initial Setting

LOCATION ON

Usage Notes

1. Displaying the location indicator overlays data in the window. If reserved lines are defined, the information overlays those lines. If not, the information overlays the scrollable data area. The data is not changed in the virtual screen and you are prohibited from typing over the location information. To view the data being overlaid, issue SET LOCATION *wname* OFF.
2. If the window is not wide enough to display the entire location indicator, the location information is truncated.

Responses

Location information for the number of lines or columns, or both, is displayed in the upper right corner of the window, using the color, highlight, and program symbol set defined for the window border. For example:

```
Lines 25 - 44 of 44
Columns 1 - 20 of 80
```

Messages and Return Codes

DMSSEF921E Window *wname* is not defined [RC=28]
DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET LOGFILE

Use the LOGFILE option to control whether or not a log file is updated.

Format

SET	LOGFILE <i>vname</i> { ON OFF } [<i>fn</i> [LOGFILE [<i>fm</i> * A1]]]]
------------	---

Operands

vname

is the name of the virtual screen.

ON

begins logging for the specified virtual screen.

OFF

discontinues logging for the specified virtual screen.

fn

is the file name of the file to which data is logged. The default file name is the name of the virtual screen.

LOGFILE

is the file type of the file to which data is logged.

fm

is the file mode of the file. The default is *, which is the first read/write disk or directory in the search order containing the specified file. See the Usage Notes for more information.

Initial Setting

LOGFILE OFF

Usage Notes

1. When the NOTYPE option is in effect for a specified virtual screen, data in the queue is not written to the virtual screen. However, the data is logged to a CMS file if logging was requested.
2. If the CMS file already exists, the data written to the virtual screen is appended to the existing CMS file. If the specified file does not exist, the file is created on the disk or directory accessed as A and the lines are inserted.
3. If you issue:
 SET LOGFILE *vname* ON
 and do not specify a file name, file type and/or a file mode, then the current values from previous settings for the log file ID are used.
4. To specify the file mode, you must also specify *fn* and LOGFILE.
5. For each full-screen session, the following line is added to the file when the first message or warning is logged:
 **** Logging started for virtual screen *vname* on mm/dd/yr at hh:mm:sec

6. If you issue HX while using full-screen CMS, data is not logged in a LOGFILE for the command or program that is executing.

Messages and Return Codes

DMSSEF921E Virtual screen *vname* is not defined [RC = 28]
DMSSET109S Virtual storage capacity exceeded [RC = 104]
DMSWVL928E Command is not valid for virtual screen *vname* [RC = 12]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET NONDISP

Use the NONDISP option to define a character for CMS and the System Product Editor that is displayed in place of nondisplayable characters.

Format

SET	NONDISP [<i>char</i>]
-----	-------------------------

Operands

char

defines a character that is displayed in place of nondisplayable characters. If not specified, a blank is used.

Initial Setting

NONDISP "

Usage Notes

1. The translation of the nondisplayable character depends upon the type of terminal, whether SET APL ON or SET TEXT ON is in effect, and the current language being used (see SET LANGUAGE).
2. Changing the NONDISP setting for CMS also changes the NONDISP setting for the System Product Editor, and vice versa.
3. Changing the NONDISP character does not change characters already displayed on the screen unless that line is altered.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
 DMSSEF926E Command is only valid on display terminal [RC = 88]
 DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET REMOTE

Use the REMOTE option to control the display of data transmissions for CMS and the System Product Editor.

Format

SET	REMOTE { ON OFF }
------------	---

Operands

ON

specifies that data is to be compressed by removing nulls and combining data when five or more of the same characters occur consecutively in a data stream. This minimizes the amount of data transmitted and shortens the buffer, thus speeding transmission.

OFF

specifies that the data stream is not to be compressed. Data is transmitted with no minimization.

Initial Setting

REMOTE ON for remote displays

REMOTE OFF for local displays.

Usage Note

Changing the REMOTE setting for CMS also changes the REMOTE setting for the System Product Editor, and vice versa.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
 DMSSEF926E Command is only valid on a display terminal [RC = 88]
 DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET RESERVED

Use the RESERVED option to specify the maximum number of lines in a window that are used to display virtual screen reserved lines.

Format

SET	RESERVED <i>wname</i> $\left\{ \begin{array}{c} rtop \\ * \end{array} \right\} \left\{ \begin{array}{c} rbot \\ * \end{array} \right\}$
------------	--

Operands

wname

is the name of the window.

rtop

is the maximum number of reserved lines that are displayed in the top of the window. The number displayed depends on the number of reserved lines defined in the virtual screen to which the window is connected, and on the number of lines in the window (see Usage Notes 2 and 3).

rbot

is the maximum number of reserved lines that are displayed in the bottom of the window. The number displayed depends on the number of reserved lines defined in the virtual screen to which the window is connected, and on the number of lines in the window (see Usage Notes 2 and 3).

Initial Setting

RESERVED *wname* * *

Usage Notes

1. Reserved lines are maintained in the virtual screen buffers and are used to display such things as title lines and PF key descriptions. Reserved line data is not scrollable and takes up space above and below the scrollable data area in the middle of the window. See the description of the DEFINE VSCREEN and WRITE VSCREEN commands for more information regarding virtual screen reserved lines.
2. The number of reserved lines displayed in the window is the MINIMUM of the number specified with the SET RESERVED command or the number defined in the virtual screen to which the window is connected. When * is specified, the number of reserved lines displayed is the number defined in the virtual screen to which the window is connected.

For example, suppose you have a window called MESSAGE that is 10 lines long. The virtual screen to which the window is connected contains 3 top reserved lines and 5 bottom reserved lines. If you issue the command:

```
set reserved message 5 2
```

when the window is displayed on the physical screen, it contains 3 top reserved lines, because the virtual screen has 3 top reserved lines, and 2 bottom reserved lines, because that is the maximum number you requested.

If you then change the size of MESSAGE so that it is now only 3 lines long, the window is displayed with 1 top reserved line and 2 bottom reserved lines, according to the rules of this note and Usage Note 3.

3. Setting reserved lines for a window is independent of the window size. The number of reserved lines to be displayed in the window is determined when the physical screen is refreshed. Lines are handled according to the following priority:
 - a. Bottom reserved lines
 - b. Top reserved lines
 - c. Data lines (top down)

Messages and Return Codes

DMSSEF921E Window *wname* is not defined [RC = 28]

DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET TEXT

SET TEXT

Use TEXT option to activate character code conversion for TEXT characters for the System Product Editor and CMS.

Format

SET	TEXT { ON OFF }
-----	--------------------

Operands

ON

activates character code conversion for TEXT characters. Before using TEXT keys, issue SET TEXT ON to ensure proper character code conversion.

OFF

specifies that no character code conversion is performed for TEXT characters and keys.

Initial Setting

TEXT OFF

Usage Notes

1. The TEXT setting is valid only when performing full-screen I/O (for example, in XEDIT or in CMS with SET FULLSCREEN ON). If you are in CP or using a line-mode terminal, SET TEXT has no effect.

If you are in CP, you can issue the TERMINAL TEXT ON command to have CP convert TEXT character codes.

2. Because the text character code conversion is costly, it is recommended that you issue SET TEXT OFF when you stop using the special text keys.
3. When SET TEXT ON is specified, APL is set OFF.
4. Changing the TEXT setting for CMS also changes the TEXT setting for the System Product Editor.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
DMSSEF524W NONDISP character reset to ".
DMSSEF926E Command is only valid on a display terminal [RC = 88]
DMSSET109S Virtual storage capacity exceeded [RC = 104]
DMSWIR329W Warning: APL/TEXT option not in effect

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SET VSCREEN

Use the VSCREEN option to indicate what action should take place when the virtual screen is updated with data.

Format

SET	VSCREEN <i>vname</i> { <table style="display: inline-table; border: none; vertical-align: middle;"> <tr> <td style="border: none; padding: 0 5px;">[</td> <td style="border: none; padding: 0 5px;">TYPe</td> <td style="border: none; padding: 0 5px;">]</td> <td style="border: none; padding: 0 5px;">[</td> <td style="border: none; padding: 0 5px;">PRotect</td> <td style="border: none; padding: 0 5px;">]</td> <td style="border: none; padding: 0 5px;">[</td> <td style="border: none; padding: 0 5px;">High</td> <td style="border: none; padding: 0 5px;">]</td> </tr> <tr> <td style="border: none; padding: 0 5px;">[</td> <td style="border: none; padding: 0 5px;">NOType</td> <td style="border: none; padding: 0 5px;">]</td> <td style="border: none; padding: 0 5px;">[</td> <td style="border: none; padding: 0 5px;">NOPRotect</td> <td style="border: none; padding: 0 5px;">]</td> <td style="border: none; padding: 0 5px;">[</td> <td style="border: none; padding: 0 5px;">NOHigh</td> <td style="border: none; padding: 0 5px;">]</td> </tr> </table> [<i>color</i>] [<i>exthi</i>] [<i>psset</i>]	[TYPe]	[PRotect]	[High]	[NOType]	[NOPRotect]	[NOHigh]
[TYPe]	[PRotect]	[High]											
[NOType]	[NOPRotect]	[NOHigh]											

Operands

vname

is the name of the virtual screen.

TYPe

specifies that data is moved to the virtual screen when the virtual screen queue is processed.

NOType

specifies that the virtual screen is not updated.

PRotect

the data is protected.

NOPRotect

the data is not protected.

High

data is displayed in high intensity.

NOHigh

data is displayed in a normal intensity.

color

the color may be Default, Blue, Red, Pink, Green, Turquoise, Yellow, or White.

exthi

the extended highlighting may be None (default), REVvideo, BLInk, or Underline.

psset

the Programmed Symbol Set (PSset) may be specified as PS0 (the default), PS1, PSA, PSB, PSC, PSD, PSE, or PSF.

Initial Setting

See DEFINE VSCREEN.

SET VSCREEN

Usage Notes

1. When NOTYPE is specified, the data is not written to the virtual screen when the queue is processed. However, the data is logged to a CMS file if logging was requested (see the SET LOGFILE command).
2. In full-screen CMS, SET VSCREEN CMS NOTYPE suppresses all updates to the CMS virtual screen. However, to suppress only the error messages from within an EXEC, use the SET CMSTYPE HT command.

Messages and Return Codes

DMSSEF921E Virtual screen *vname* is not defined [RC=28]
DMSSEF928E Command is not valid for virtual screen *vname* [RC=12]
DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET WINDOW

Use the WINDOW option to specify

- whether a window is variable or fixed size
- if the window is affected when the virtual screen that the window is showing is updated.
- whether or not the window qualifies as the topmost window.

Format

SET	WINDOW <i>wname</i> { [VARi able] [POP] [TOP] } [FIX ed] [NO Pop] [NO Top] }
------------	--

Operands

wname

is the name of the window.

VARiable

indicates that the current number of lines in the window may vary from 0 to the number of lines defined for the window, depending on how much data is displayed,

FIXed

indicates that the number of lines in the window is always constant.

POP

specifies that the window is displayed on top of all other windows when the virtual screen that the window is showing is updated.

NOPop

specifies that there is no effect on the window's position in the ordered list of windows when the virtual screen that the window is showing is updated.

TOP

specifies that the window may qualify as the topmost window. Most windowing commands process the topmost window by default or when = is specified as the window name.

NOTop

specifies that the window cannot qualify as the topmost window. Windows defined as NOTOP are not processed by default or when a command is specified with = for the window name.

Initial Setting

See DEFINE WINDOW.

SET WINDOW

Messages and Return Codes

DMSSEF921E Window *wname* is not defined [RC=28]

DMSSET109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

SET WMPF

Use the WMPF option to set a WMPF key to execute a windowing command.

Format

SET	WMPF <i>nn</i> [<i>pseudonym</i>] [<i>keyword</i>] <i>string</i>]
	[NOWRITE] [DELAYED]

Operands

nn

is a number from 1 to 24 indicating which PF key is being set.

pseudonym

is a 9-character representation of the PF key definition. The pseudonym is displayed in the PF key definition area at the bottom of the CMS window. The pseudonym may be up to nine characters in length.

NOWRITE

suppresses overwriting of the PF key pseudonym when you set a WMPF key.

keyword

indicates when the command associated with the PF key is executed in relation to commands entered at the terminal. The *keyword* may be one of the following:

DELAYED

delays the execution of the command string. When the key is pressed, the command string is displayed in the input area and is not executed until you press the ENTER key. If anything is currently in the input area, it is overlaid and no commands entered on the physical screen are processed. This is the default setting if no keyword is specified on the SET WMPF command.

ECHO

executes the command immediately when the program function key is pressed. The key definition is echoed above the command line in the WM window.

NOECHO

executes the command immediately when the program function key is pressed. The key definition is not echoed on the physical screen.

Note: When a WMPF key is set to RETRIEVE the keyword is ignored.

string

is the command(s) to be executed when the key is pressed.

Initial Setting

WMPF 01 Help	NOECHO	HELP
WMPF 02 Top	NOECHO	SCROLL TOP =
WMPF 03 Quit	NOECHO	DROP WINDOW WM
WMPF 04 Clear	NOECHO	CLEAR WINDOW =
WMPF 05 Copy	NOECHO	PUT SCREEN COPY SCREEN

SET WMPF

WMPF 06 Retrieve		RETRIEVE
WMPF 07 Backward	NOECHO	SCROLL BACKWARD = 1
WMPF 08 Forward	NOECHO	SCROLL FORWARD = 1
WMPF 09 Maximize	NOECHO	MAXIMIZE WINDOW =
WMPF 10 Left	NOECHO	SCROLL LEFT = 10
WMPF 11 Right	NOECHO	SCROLL RIGHT = 10
WMPF 12 Restore	NOECHO	RESTORE WINDOW =

Note: These are initial settings. On a terminal equipped with 24 PF keys, PF 13 through 24 have the same values as PF keys 1 through 12, respectively.

Usage Notes

1. You can set a WMPF key to execute any of the following commands:

CLEAR WINDOW	PUT SCREEN	SCROLL
CP	QUERY BORDER	SET BORDER
DROP WINDOW	QUERY HIDE	SET LOCATION
HELP	QUERY LOCATION	SET RESERVED
HIDE WINDOW	QUERY RESERVED	SET WINDOW
MAXIMIZE WINDOW	QUERY SHOW	SET WMPF
MINIMIZE WINDOW	QUERY WINDOW	SHOW WINDOW
POP WINDOW	QUERY WMPF	SIZE WINDOW
POSITION WINDOW	RESTORE WINDOW	

In the WM environment, you can enter **HELP** (WMPF 1) to see the list of commands that are available. The WM environment creates a **WMHELP** window and **WMHELP** virtual screen to display the list. To exit the WM window, use the **DROP WINDOW** command (WMPF 3).

2. To cancel a PF key definition, enter:
SET WMPF nn
3. When you press a PA key or a WMPF key in the WM window and the key that was pressed does not update the command line, then input on the command line is rewritten. The next time you press enter it is executed.
4. The **RETRIEVE** function saves previously entered commands in a buffer that is 256 characters long. When you enter the WM environment, the buffer contains an asterisk (comment), and commands are added to the buffer as they are entered until it is full. As you continue to enter commands, the oldest commands are deleted and the most current commands are added.

Pressing the PF key assigned to **RETRIEVE** displays the next command in the buffer on the command line. Each time you press the key, the previously entered command is displayed until the oldest one is reached. Then, **RETRIEVE** returns the most current command. Once the command is on the command line, press enter to execute it. You may also modify the command, then press enter to execute the new command.

5. The **NOWRITE** option is particularly useful when you have changed the bottom reserved area in the WM virtual screen and you do not want the area overwritten when you set a WMPF key. However, when you enter the WM environment for the first time, the WMPF key definitions are overwritten in the bottom reserved area of the VM virtual screen.

Messages and Return Codes

DMSSEF109S Virtual storage capacity exceeded [RC = 104]
DMSSEF525E Invalid PFkey number [RC = 24]
DMSSEF926E Command is only valid on a display terminal [RC = 88]
DMSSET109S Virtual storage capacity exceeded [RC = 104]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SHOW WINDOW

Use the SHOW WINDOW command to place a window at the top of the window display order and to connect the window to a virtual screen.

Format

SHOW WINDOW	<i>wname</i> [ON <i>vname</i> [<i>line col</i>]]
--------------------	--

Operands

wname

is the name of the window.

vname

is the name of the virtual screen to which the window is connected.

line

is the line number of the virtual screen where the upper left corner of the window is placed.

col

is the column number of the virtual screen where the upper left corner of the window is placed.

Usage Notes

- Multiple windows may be connected to a single virtual screen.
- If the window is already connected to a virtual screen when you issue the SHOW WINDOW command, you do not have to specify the virtual screen information.
- When you specify a virtual screen name, line, and column, the line and column values must be less than or equal to the corresponding virtual screen dimensions. The minimum line and column value is 1. If line and column are not specified, the default is 1 for both. If the line specified is past the current virtual screen bottom, the window is connected to the virtual screen bottom.
- A variable size window is only displayed when there is at least one scrollable line to show. If a variable size window is showing a virtual screen that does not contain any scrollable lines, the SHOW WINDOW command places the window at the top of the window display order and connects it to the specified virtual screen, but it is not displayed.
- When you are using full-screen CMS and you enter the SHOW WINDOW command from the command line, the command is executed and then the screen is refreshed. As part of the refresh processing, any pop-type window that has output waiting is moved to the top of the order of displayed windows. Therefore, the window that you specified may not be displayed at the top of the display order because another has been popped afterwards.

Messages and Return Codes

DMSSHO386E	Missing operand(s) [RC=24]
DMSSHO388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSSHO389E	Invalid operand: <i>operand</i> [RC=24]
DMSSHO391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSSHO921E	Window <i>wname</i> is not defined [RC=28]
DMSWMM921E	Virtual screen <i>vname</i> is not defined [RC=28]

DMSWMM921E Window *wname* is not defined [RC=28]

DMSWMM923E Specified location is outside the virtual screen [RC=32]

DMSWMM929E Window *wname* is not connected to a virtual screen [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

SIZE WINDOW

Use the SIZE WINDOW command to change the number of lines and columns for a specified window.

Format

SIZE WINDOW	$\left\{ \begin{array}{c} wname \\ = \end{array} \right\} \text{ lines } [cols]$
--------------------	--

Operands

wname

is the name of the window. An "=" indicates that the size of the topmost window is changed.

lines

is the number of lines in the window.

cols

is the number of columns in the window. The default is the current number of columns.

Usage Note

1. The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.

Messages and Return Codes

DMSSIZ386E	Missing operand(s) [RC=24]
DMSSIZ388E	Invalid keyword: <i>keyword</i> [RC=24]
DMSSIZ389E	Invalid operand: <i>operand</i> [RC=24]
DMSSIZ391E	Unexpected operand(s): <i>operand</i> [RC=24]
DMSSIZ921E	Window <i>wname</i> is not defined [RC=28]
DMSSIZ922E	Window does not fit entirely on the screen [RC=32]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

WAITREAD VSCREEN

Use the WAITREAD VSCREEN command from an exec to update the virtual screen with data in the virtual screen queue, refresh the physical screen, and wait for the next attention interrupt.

Format

WAITREAD VScreen	<i>vname</i>
-------------------------	--------------

Operands

vname
is the name of the virtual screen that is waiting for input.

Usage Notes

1. All windows showing virtual screens other than the virtual screen specified in the WAITREAD command are protected. The windows showing the specified virtual screen are either protected or unprotected, depending on how the data being displayed is defined in the virtual screen.
2. WAITREAD VSCREEN executes in the following manner:
 - Each virtual screen is updated with data from the virtual screen queue
 - The screen image is rebuilt based on the ordered list of windows and displayed on the physical screen
 - Wait for the next interrupt (The virtual screen specified in the WAITREAD VSCREEN command is now active.)
 - When the interrupt is received, the screen is read
 - Information regarding the key pressed, the cursor, and modified fields are passed back to the exec in variables.

The virtual screen can then be updated with the variables by using the WRITE VSCREEN command.

The exec variables contain:

WAITREAD.0

the number of variables returned (excluding WAITREAD.0)

WAITREAD.1

the key that caused the attention interrupt (such as PAKEY n, PFKEY n,CLEAR, or ENTER)

WAITREAD.2

the word CURSOR followed by the line number and column number of the cursor in the virtual screen, followed by the word DATA or RESERVED indicating the area that the cursor was located. If the cursor was in a DATA or RESERVED area, the following line and column number pairs may be returned:

	LINE	COLUMN
a.	n	n
b.	0	n
c.	0	2

where:

- a. means that the cursor was positioned between the top and bottom of the virtual screen. The line number may range from one (1) to the number of scrollable data lines in the virtual screen. The column number may range from one (1) to the number of columns in the virtual screen.
- b. means that the cursor was positioned on the line immediately following the bottom of the virtual screen. The line number will be zero (0), and the column number may range from one (1) to the number of columns in the virtual screen.
- c. means that the cursor was positioned on a line following the bottom of the virtual screen, but that the line was not necessarily the one immediately following the virtual screen bottom. The line number will be zero (0), and the column number will be two (2).

If the cursor was not in a DATA or RESERVED area, the line and column numbers will both be -1.

WAITREAD.3

WAITREAD.n

where each variable contains the word DATA or RESERVED indicating the type of text updated, followed by the line number and column number of the field that was modified, followed by the modified text.

Note: The keyword values returned in WAITREAD.0, WAITREAD.1, and WAITREAD.2, as well as the EXEC variables WAITREAD.n, are always in American English (AMENG).

3. WAITREAD VSCREEN is an important command for EXECs that read from and write to windows. A typical sequence for such an exec would be:
 - Define a window and virtual screen (DEFINE WINDOW and DEFINE VSCREEN commands)
 - Connect the window to the virtual screen (SHOW WINDOW command)
 - Write data to the virtual screen (WRITE VSCREEN command)
 - Issue the WAITREAD VSCREEN command

When an interrupt is received, the exec can process the WAITREAD.n variables and update the virtual screen using the WRITE VSCREEN command.

4. A field definition character is placed at the start of each row if:
 - A window is not connected to a virtual screen at column 1, or
 - The number of columns in the window, virtual screen, and physical screen are not the same.

As a result, data is shifted one character to the right in each row so that data in column one is not lost. Lines may be truncated on the right, and the location information indicates that the window is showing one less character from the virtual screen.

5. When windows overlap on the physical screen, a field definition character is placed at each window boundary.
6. If only part of a field from a virtual screen is displayed on the physical screen and the field is modified, the entire field is returned as a modified field in a variable WAITREAD.n.
7. The same field can be modified in different windows. These modifications are processed from the top of the screen to the bottom, moving from left to right.
8. The lines in a window that are not top reserved lines, bottom reserved, or data lines from the virtual screen are called pad lines. When a window is connected to the active virtual screen, the pad lines are unprotected. If there are multiple windows connected to the same active virtual screen, only the pad lines of the top-most window are unprotected.
9. Any part of a window that does not map to the virtual screen is protected. For example, suppose you have defined a window that is 24 rows by 80 columns and a virtual screen that is 20 rows by 60 columns. When you connect the window to the virtual screen at row 1 column 1, the 4 rows and 20 columns in the window that do not map to the virtual screen are protected.

Messages and Return Codes

- DMSWIO494W FULLREAD set off.
- DMSWIO614E Screen modifications lost. See 'SET FULLREAD' to use PAkeys safely.
- DMSWIO629W Screen modifications may be lost. Press ENTER key to process screen changes.
- DMSWIR329W Warning: APL/TEXT option not in effect
- DMSWIR696W Invalid data received from the display
- DMSWRD386E Missing operand(s) [RC=24]
- DMSWRD388E Invalid keyword: *keyword* [RC=24]
- DMSWRD391E Unexpected operand(s): *operand* [RC=24]
- DMSWRD622E Insufficient free storage [RC=104]
- DMSWRD631E WAITREAD can only be executed from an EXEC-2 or REXX EXEC [RC=4]
- DMSWRD917E No windows are showing virtual screen *vname* [RC=4]
- DMSWRD921E Virtual screen *vname* is not defined [RC=28]
- DMSWRD925E I/O error on screen [RC=100]
- DMSWRD926E Command is only valid on a display terminal [RC=88]
- DMSWRD928E Command is not valid for virtual screen *vname* [RC=12]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

WAITT VSCREEN

Use the WAITT VSCREEN command to update a virtual screen with data.

Format

WAITT VScreen	[<i>vname</i> * _]
---------------	----------------------------

Operands

vname

is the name of the virtual screen to be updated. An * indicates that all the virtual screens are updated. An _ is the default.

Usage Notes

1. WAITT VSCREEN updates the virtual screen with any data that has been written to it. The data is moved from the queue to the virtual screen data buffer. The following also may occur:
 - If logging is requested, the data is appended to the virtual screen log file.
 - If NOTYPE is specified (or is in effect for the virtual screen), then the data is logged (if logging was requested), and then discarded.
2. Queuing is handled so that applications need not be concerned with the virtual screen size and the location of the data in the virtual screen buffers. When the screen is refreshed, the queued writes are moved into the virtual screen with one field per line of output.

When the queue becomes full, writing stops until a window connected to a virtual screen is cleared or scrolled. As you scroll forward, the oldest lines that have been scrolled are deleted, new lines are appended at the bottom, and the lines are renumbered. (Lines that have not been scrolled are not deleted.)
3. When the virtual screen is updated, any windows defined as POP showing the virtual screen are popped to the top of the ordered list of windows.
4. In full-screen CMS, when the virtual screen being updated is not the CMS output virtual screen, the DMSWVT messages are usually displayed after the echo and before the other messages.

Messages and Return Codes

DMSWAT386E Missing operand(s) [RC=24]
 DMSWAT388E Invalid keyword: *keyword* [RC=24]
 DMSWAT391E Unexpected operand(s): *operand* [RC=24]
 DMSWAT921E Virtual screen *vname* is not defined [RC=28]
 DMSWAT928E Command is not valid for virtual screen *vname* [RC=12]

Messages while logging and/or updating virtual screen data to disk or directory:

DMSFNS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
 DMSFNS1252T Rollback unsuccessful for file pool *filepoolid*
 DMSSTT062E Invalid character *char* in fileid *fn ft fm*

DMSWVT037E Filemode *fm* is accessed as read/only
 DMSWVT069E Filemode *mode* not accessed
 DMSWVT105S Error *nn* writing file *fn ft fm* on disk or directory
 DMSWVT109S Virtual storage capacity exceeded
 DMSWVT531E Disk or file space is full; set new filemode or clear some space
 DMSWVT924E Data was truncated
 DMSWVT933W Logging stopped for virtual screen *vname*
 DMSWVT934E Text was not written to virtual screen. No field was defined.
 DMSWVT1258E You are not authorized to write to file *fn ft fm|dirid*
 DMSWVT1262S Error *nn* opening file *fn ft fm*

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

Return Codes

- 0 Normal execution
- 13 Virtual screen full

WRITE VSCREEN

Use the WRITE VSCREEN command to enter information in a virtual screen. Information is queued to a virtual screen and is displayed the next time the screen is refreshed.

The WRITE VSCREEN command provides many possibilities for defining fields, writing data, and updating the plane buffers associated with the virtual screen (color, extended highlighting, and Programmed Symbol Sets).

Format

WRITE VScreen	<p><i>vname line col length</i> [([REServed] [optionA] [optionB] [optionC] [optionD] [])]</p> <p>OptionA: [BLANKs] [NULs]</p> <p>OptionB: [PROtect] [High] [NOPROtect] [NOHigh] [Invisible]</p> <p>OptionC: [color] [exthi] [pssset]</p> <p>OptionD: { FIELD } { DATA } { COLOR } <i>text</i> { EXTHI } { PSS }</p> <p><i>Note: If option D is used, a right parenthesis should not be used to mark the end of the options.</i></p>
----------------------	--

Operands

vname

is the name of the virtual screen.

line

is the line number of the virtual screen where the write is to begin.

col

is the column number of the virtual screen where the write is to begin.

length

is the length to be written.

REServed

indicates that the line number refers to a reserved line. If you specify RESERVED, *line* must be a number less than or equal to the size of the reserved area. A negative line number indicates a write in the bottom reserved area, and a positive line number indicates a write in the top reserved area.

Option A

Option A is used for padding text in the data buffer. The following may be specified:

BLAnks

pad data with blanks

NULLs

pad data with nulls (default)

Option B

Option B is the attribute for defining a field. The following may be specified:

PROtect

protected field

NOProtect

not protected field

High

high intensity field

NOHigh

normal intensity field

Invisible

invisible field. The data written is not displayed on the screen.

Option C

Option C is the extended attribute for updating the virtual screen buffers and/or for padding the text being written. If option C is not specified, then the default characteristics of the virtual screen are used, if necessary (see the DEFINE VSCREEN command). The following may be specified:

color

is the color of the field. It may be DEFault, Blue, Red, Pink, Green, Turquoise, Yellow, or White.

exthi

is the extended highlighting of the field. It may be None, REVvideo, BLInk, or Underline.

psset

is the Programmed Symbol Set of the field. It may be PS0, PS1, PSA, PSB, PSC, PSD, PSE, or PSF. The psset must already be loaded in the display. If not, no psset is used.

Option D

Option D is the buffer for the text. The following may be specified:

FIELD

creates a field. All the buffers will be initialized with either the virtual screen defaults, or the settings specified in options A and/or B and/or C. Text is placed in the field left justified, and is padded or truncated to the field length.

DATA

indicates that the text is to be written to the data buffer of the virtual screen. The text is written for the length specified or until the next field is encountered.

COLOR

indicates that the text is composed of color codes that are to be written to the color buffer. The text is written for the length specified or until the next field is encountered. Use the following characters to represent the colors:

- 1 Blue
- 2 Red
- 3 Pink
- 4 Green
- 5 Turquoise
- 6 Yellow
- 7 White
- 0 Defaults to the color of the field

EXTHI

indicates that the text is composed of extended highlighting codes that are to be written to the extended highlight buffer. The text is written for the length specified or until the next field is encountered. The following characters represent the extended highlighting:

- 0 Defaults to the extended highlight of the field
- 1 Blink
- 2 Revvideo
- 4 Underline

PSS

indicates that the text is composed of Programmed Symbol set codes that are to be written to the PSset buffer. The text is written for the length specified or until the next field is encountered. The following characters represent the PSS:

0 (defaults to the PSS of the field) 1 A B C D E F

Note: The letters must be specified in upper case.

text

is the text to be written.

Usage Notes

1. When WRITE VSCREEN is issued, the text and its characteristics are held in the virtual screen queue created by DEFINE VSCREEN. To move the text from the queue to the virtual screen, you should issue either WAITREAD VSCREEN, WAITT VSCREEN, or the REFRESH command. This process is required when issuing the WRITE VSCREEN command from an EXEC.
2. Use the INVISIBLE option to prevent data such as passwords from being displayed on the screen.
3. When a field is defined, the first character contains the start field. The start field is a one-byte character identifying the attributes for the field. The start field character is protected and cannot be written to. If option D is not specified, the default is FIELD.

For more information on fields, refer to the *IBM 3270 Information Display System Data Stream Programmer's Reference, GA23-0059*.

4. For color and extended highlighting in a DBCS string, the first byte of a double-byte character determines the attributes for both bytes. You cannot specify character attributes for PSS within a DBCS string.

5. The column operand must be greater than or equal to zero. Specifying a column number of zero is valid only when writing to the scrollable area, in which case it is equivalent to specifying column number one. When writing a field, a start field is placed in the column specified. The text always begins in the next column. When writing COLOR, EXTHI, PSS, or DATA, the text begins in the column specified, or in the next available column after the start field(s).
6. Specifying a line number of zero is valid only when writing to the scrollable area. When specifying a line number of zero and writing a field, a field is created at the line past the current bottom of the virtual screen. This provides a means for writing sequentially to the virtual screen. When doing sequential writes, a field always fills an entire line. The length of the field is determined as follows (assume a virtual screen with 80 columns):

- If the length specified is zero, as in the example:

```
write vscreen cms 0 0 0 (field Enter your name:
```

A field is created at the line past the current bottom of the CMS virtual screen. A start field is placed in column 1 and the text begins in column 2. All the text is written and the length of the field is set to 80.

- If length specified is less than the number of columns in the virtual screen, as in the example:

```
write vscreen cms 0 0 10
```

A field is defined at one line past the current bottom of the virtual screen. Even though the length was specified as 10, the length of the field is set to 80 in order to fill the entire line.

- If length specified is greater than the number of columns in the virtual screen, as in the example:

```
write vscreen cms 0 0 100
```

A field is defined at one line past the current bottom of the virtual screen. In this case, the length of the field is set to 160 and the field fills two lines.

7. When using Option D, SET CHARMODE must be ON to update COLOR, EXTHI, and PSS. If SET CHARMODE is OFF they are ignored. Switching from SET CHARMODE ON to OFF may produce some undesirable results, such as a field having attributes that you intended only for a character.
8. When specifying a line number of zero and writing COLOR, EXTHI, PSS, or DATA, you are acting on the "current field." The current field is the most recent field written to the scrollable data area of the virtual screen. The column number specifies the position in the current field. For example:

```
write vscreen cms 0 5 0 (COLOR 11116666777
```

will update the color buffer starting at the fifth position of the current field.

9. If the length specified is less than the length of the text, then the text is truncated. If the length specified is greater than the length of the text, then the text is padded with the characteristics specified in option A and/or option B and/or option C. If the options are not specified, then text is padded with the characteristics defined for the virtual screen.

Note that when you are not writing a field, the text is written for the length specified or until the next field is encountered.

For example:

WRITE VSCREEN

```
write vscreen cms 1 1 20 (blank field Enter your name:
write vscreen cms 0 0 20 (red color 11111
```

writes "Enter your name: ." The "Enter" is displayed blue and "your name:" is displayed red.

The example:

```
write vscreen cms 1 1 10 (data Enter your name:
```

writes "Enter your."

10. When defining a virtual screen, the top reserved area is defined as one continuous field and the bottom reserved area is defined as one continuous field. However, the scrollable area is not defined as a field. To write to the scrollable data area, you must define a field.
11. In a virtual screen, the lines in the top reserved area are numbered starting from the top. The top line is 1, the second line is 2, etc. In the bottom reserved area, lines are numbered starting at the bottom and have negative values. The bottom line is -1, the second line up is -2, the third line up is -3, etc.
12. Once a field is defined, you cannot change the attributes (PROtect, NOPROtect, High, NOHigh) without redefining the field. However, you can change the extended attributes (color, exthi, pset) or data of the field. For example, suppose you define the following field:

```
write vscreen cms 5 20 0 (field Enter your name:
```

To change the color extended attribute, you can enter:

```
write vscreen cms 5 20 0 (COLOR 222224444455555
```

As a result, beginning at line 5, column 21 (because there is a start field character in column 20), "Enter" will be red, "your" will be green, and "name:" will be turquoise.

13. When writing sequentially (see Usage Note 6) and the text contains a character assigned to X'15' (end-of-line, or EOL) via the SET INPUT command, that character indicates a line end. The text following it is continued on the next line. A new line is written for each EOL.
14. You should load programmed symbol sets prior to using XEDIT or prior to issuing the SET FULLSCREEN or SET FULLSCREEN RESUME commands. This ensures that the programmed symbol sets are available when you use full-screen CMS or XEDIT.

For line-mode CMS, the programmed symbol sets are detected the first time you display a window or when you invoke XEDIT. If you load programmed symbol sets after you have displayed a window or after you have invoked XEDIT, you must invoke XEDIT again to ensure detection of the new programmed symbol sets.

Messages and Return Codes

DMSWRT622E Insufficient free storage [RC=104]
DMSWRT921E Virtual screen *vname* is not defined [RC=28]
DMSWRT923E Specified location is outside the virtual screen [RC=32]
DMSWRT928E Command is not valid for virtual screen *vname* [RC=12]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

Border Commands

Border commands are single-character commands for windowing that you may enter in the corners of the border.

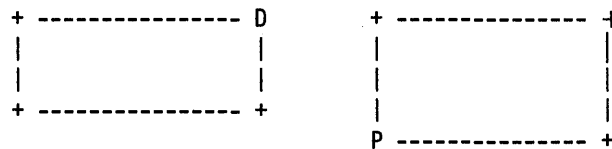
The commands are processed as soon as they are entered.

The border commands are:

- B** Scrolls the window backward
- C** Clears the window of scrollable data
- D** Drops the window
- F** Scrolls the window forward
- H** Hides the window
- L** Scrolls the window to the left
- M** Changes the location of the window
- N** Minimizes the window
- O** Restores the window
- P** Pops the window
- R** Scrolls the window to the right
- S** Changes the size of the window
- X** Maximizes the window

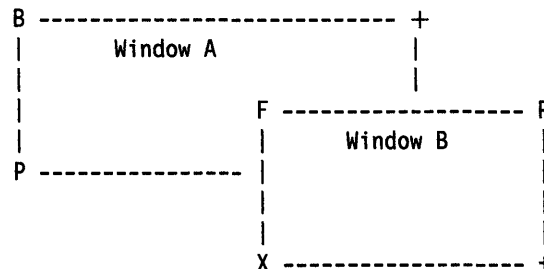
General Usage Notes

1. Use the SET BORDER command to control the border around a window. The borders are ON by default.
2. The border commands can be placed in any of the four corners of the window border. For example:



are valid ways to enter border commands. You can type multiple border commands in several windows before pressing the ENTER key.

3. When multiple commands are entered, the order of execution is from left-to-right and top-to-bottom on the physical screen regardless of the order of the windows. For example:



executes command B first, then command F, then command R, then command P, and finally command X.

General Error Messages

DMSWBX931E Invalid border command: *character*

For a general discussion and examples of how to use border commands, see the *VM/SP CMS User's Guide*.

B

Use the B border command to scroll the window backward.

Format

B	
----------	--

Usage Notes

1. If the window has been cleared (see the C Border command), the B command scrolls you to the bottom of the virtual screen.
2. If the window is positioned in the middle of the virtual screen and you scroll backward using the B command which would result in scrolling past the top, the window is repositioned at the virtual screen top and stops. If the window is positioned at the virtual screen top and you scroll backward, the window is repositioned at the virtual screen bottom and stops.

C

Use the C border command to clear the window of all scrollable data.

Format

C	
----------	--

Usage Notes

1. If the window you want to clear is variable size, the window is not displayed when the screen is refreshed.
2. The C command has no effect if the window is displaying a virtual screen which was defined with less than two data lines. In addition, in a System Product Editor session, you cannot clear the window that the System Product Editor is using.

D

Use the D border command to place a window beneath all other windows.

BORDER COMMANDS

Format

D	
----------	--

Usage Note

When using full-screen CMS, you cannot drop a window that is showing the virtual screen indicated in the status area message. For example, if the CMS window is showing the CMS virtual screen, and the status area message instructs you to "Scroll for more information in vscreen CMS," you cannot drop the CMS window. You can drop any other window.

F

Use the F border command to scroll the window forward.

Format

F	
----------	--

Usage Notes

1. If the window has been cleared, the F command scrolls the window to the top of the virtual screen.
2. If the window is positioned in the middle of the virtual screen and you scroll forward using the F command which results in scrolling past the bottom, the window is repositioned at the virtual screen bottom and stops. If the window is positioned at the virtual screen bottom and you scroll forward, the window is cleared. A subsequent scroll forward positions the window at the virtual screen top. If the window being scrolled is variable size and it is positioned at the bottom of the virtual screen, the window is not displayed at the next refresh when it is scrolled forward (see the CLEAR WINDOW command).

H

Use the H border command to hide the window.

Format

H	
----------	--

L

Use the L border command to scroll the window left two-thirds the size of the window or to the left edge of the virtual screen to which the window is connected.

Format

L	
---	--

Usage Note

The window is scrolled up to a maximum of two-thirds the width of the window. If you try to scroll beyond column 1 of the virtual screen, the window is placed in column 1 of the virtual screen.

M

Use the M border command to move the window corner where the command was typed to the location of the cursor when the interrupt occurred.

Format

M	
---	--

Usage Note

The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.

Error Messages

DMSWBX922E Window does not fit entirely on the screen

N

Use the N border command to minimize the window.

Format

N	
---	--

Usage Note

The O command returns the window to its size and location prior to the minimize.

O

Use the O border command to restore a maximized or minimized window to its original size and location.

BORDER COMMANDS

Format

O	
---	--

P

Use the P border command to place a window on top of all other windows.

Format

P	
---	--

R

Use the R border command to scroll the window right two-thirds the size of the window or to the right edge of the virtual screen to which the window is connected.

Format

R	
---	--

Usage Note

The window is scrolled up to a maximum of two-thirds the width of the window. If you try to scroll beyond the last column of the virtual screen, the window is scrolled to the last column of the virtual screen.

S

Use the S border command to change the size of the window based on the position of the cursor. The corner where the command was typed is moved to the location of the cursor when the interrupt occurred.

Format

S	
---	--

Usage Note

The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.

Error Messages

DMSWBX922E Window does not fit entirely on the screen
 DMSWBX930E Cursor is not in a valid location

X

Use the X border command to maximize a window.

Format

X	
----------	--

Usage Notes

1. The O command returns the window to its size and location prior to the maximize.
2. A maximized window is positioned at line 1, column 1 of the physical screen.
3. A variable size window that is maximized still retains its variable size properties. Thus, depending on how many lines exist in the virtual screen to which the window is connected, the window may appear to be less than full screen size when it is displayed on the physical screen.

 For example, if a variable size window is connected to line 1 of a virtual screen which contains three data lines, when it is maximized:
 - it moves to line 1, column 1; or line -1, column 1, of the physical screen;
 - it's width is the size of the physical screen; and,
 - it contains only three data lines.
4. When you maximize a window so that it fills the entire screen and covers all other windows, you may not be able to enter commands. The WM window is automatically displayed, and the WMPF keys and command line are available to manipulate the window. Use the RESTORE WINDOW command to restore the window and the DROP WINDOW command to exit the WM environment.

Chapter 4. Special Commands Used in Command Environments

There are four commands that make use of the lists displayed by the command environments. They can only be used in the environments listed below; they cannot be entered from the CMS command line. They are:

Command	Command Environment
ALIALIST	FILELIST
AUHLIST	DIRLIST FILELIST
DISCARD	DIRLIST FILELIST MACLIST PEEK RDRLIST
EXECUTE	CSLLIST DIRLIST FILELIST MACLIST RDRLIST

ALIALIST

Use the ALIALIST command only in the FILELIST environment. PF 9 is assigned to ALIALIST on the FILELIST screens for the SHARE and SEARCH option.

The alias information that is displayed depends on whether the file the ALIALIST command is issued against is:

- a base file belonging to you
- a base file belonging to another user
- an alias.

If you issue ALIALIST for a base file that you own, the following information will be listed:

- user IDs of those having an alias of the base file
- the number of aliases they have
- the file identifier of the alias, unless the alias is in another user's directory structure and you do not have read or write authority to that directory.

If you issue ALIALIST for a base file that another user owns, the file names of the aliases that you have to the base file are displayed.

If you issue ALIALIST for an alias, the owner of the base file is displayed. The base file name is also listed unless you do not have read or write authority to the directory containing the base file. The same information is displayed for erased aliases and revoked aliases. No base file is displayed for erased aliases since the base file no longer exists.

The information is placed in an XEDIT file named *userid* ALIALIST A0.

Format

ALIALIST	<i>fn ft dirid</i> [(options...[])]
	<u>Options:</u> [REFRESH]

Operands

fn ft
is the name of the base file or alias.

dirid
is the name of the directory that contains the base file or alias.

Option**REFresh**

updates the display of alias information with any changes made to alias information since the last update.

Usage Notes

- Entering CMS commands from ALIALIST:

Begin CMS commands with “CMS” to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.

- Default Key Settings for ALIALIST

Entering the ALIALIST command executes the PROFALIA XEDIT macro. It sets the keys to the following values (note that the ALIALIST command is only valid for files in SFS directories):

Key	Setting	Action
PF 1	Help	Displays the ALIALIST command description.
PF 2	Refresh	Updates the list to display new or deleted alias information.
PF 3	Return	Return to the FILELIST screen.
PF 4	Sort (type)	Sort the list by file type, file name.
PF 5	Sort (name)	Sort the list of files by file name, file mode.
PF 6	Sort (dir)	Sort the list by directory name, file name, file type.
PF 7	Backward	Scroll backward one screen.
PF 8	Forward	Scroll forward one screen.
PF 9	S(user)	Sort the list by user ID.
PF 10		Unassigned.
PF 11		Unassigned.
PF 12		Unassigned.

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFALIA XEDIT macro sets synonyms that you can use to sort your ALIALIST screen. The synonyms are:

SNAME Sorts the list alphabetically by file name, file type.

STYPE Sorts the list alphabetically by file type, file name.

SDIR Sorts the list by directory name.

SUSER Sorts the list by user ID, file name, and file type.

Example

In this example, the owner of the base file CMSFILES SCRIPT places the cursor on the line containing CMSFILES SCRIPT in either the FILELIST screen with the SHARE or SEARCH option, and presses PF 9.

```

GRASSI  ALIALIST  A0 V 190 Trunc=190 Size=4 Line=1 Col=1 Alt=0
Base file = CMSFILES SCRIPT FPOOL1:GRASSI.GOODIES.FOOD
Userid  Num Filename Filetype Directory
GRASSI  1 CMSFILES SCRIPT .SALES
SMITH   1 FILES SCRIPT .
STONE  1 CMSFILES SCRIPT .GOODIES
STONE  2

1= Help      2= Refresh 3= Return    4= Sort(type) 5= Sort(name) 6= Sort(dir)
7= Backward 8= Forward 9= S(user) 10=          11=          12=

====>
X E D I T 1 File
    
```

Figure 22. Sample ALIALIST Screen

Messages and Return Codes

- DMSWAL651E ALIALIST must be issued from FILELIST [RC=40]
- DMSWAL653E Error executing QUERY ALIAS, rc = nn [RC = nn]
- DMSWAL1230E ALIALIST is invalid for minidisk file [RC=00]
- DMSWAL1231E ALIALIST is invalid on a directory [RC=00]
- DMSWAL1233E Invalid use of REFRESH option [RC=40]
- DMSWAL9059E No aliases for this base file [RC=00]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811

AUTHLIST

The AUTHLIST command may only be used in the FILELIST and DIRLIST environments. Use the AUTHLIST command to display authority information in a full screen environment. PF 6 is assigned to the AUTHLIST command on the DIRLIST screen and on the FILELIST screen for the SHARE and SEARCH options.

AUTHLIST lists the authorities that you have for a specified file or directory. If you are the owner of the specified file or directory, the AUTHLIST command also lists the users who have been granted read or write authority to the file or directory and the authority that they have.

The information is placed in an XEDIT file named *userid* AUTHLIST A0.

Format

AUTHlist	[<i>fn ft</i>] <i>dirid</i> [(options... [])]]
	<u>Options:</u> [REFresh]

Operands

fn ft
is the name of the file.

dirid
is the directory name where the specified file is contained. If *fn* and *ft* are not specified, then *dirid* is the name of the directory for which you want authorities displayed.

Option

REFresh
updates the display with any changes to the authority information since the last update.

Usage Notes

1. Entering CMS commands from AUTHLIST

Begin CMS commands with "CMS" to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.

2. Default Key Settings for AUTHLIST

Entering the AUTHLIST command executes the PROFAUTH XEDIT macro. It sets the keys to the following values (note that the AUTHLIST command is only valid for files in SFS directories):

AUHLIST

Key	Setting	Action
PF 1	Help	Displays the AUHLIST command description.
PF 2	Refresh	Updates the list to display new or changed authority information.
PF 3	Return	Return to the FILELIST, or DIRLIST screen.
PF 4	S(Grantee)	Sort the list by grantee.
PF 5	Sort (W)	Sort the list by write authority and grantee.
PF 6		Unassigned.
PF 7	Backward	Scroll backward one screen.
PF 8	Forward	Scroll forward one screen.
PF 9		Unassigned.
PF 10		Unassigned.
PF 11		Unassigned.
PF 12		Unassigned.

Note: On a terminal equipped with 24 PF keys, PF keys 13 to 24 are assigned the same values as PF keys 1 to 12 as discussed here.

In addition to setting the above PF keys, the PROFAUTH XEDIT macro sets synonyms that you can use to sort your AUHLIST screen. These synonyms are:

SGRANTEE Sorts the list alphabetically by grantee.

SWRITE Sorts the list by write authority.

3. File system administrators have write authority to all files and directories, although their user IDs do not appear in the list.
4. If an object is externally protected through an External Security Manager (ESM), the listed authorities may not be correct. Users may be authorized who do not appear in the list.

Example

In this example, the owner of the base file CMSFILES SCRIPT places the cursor on the line containing CMSFILES SCRIPT in either the FILELIST screen with the SHARE or SEARCH option, and presses PF 6.

```

GRASSI  AUTHLIST A0 V 165 Trunc=165 Size=4 Line=1 Col=1 Alt=0
File = CMSFILES SCRIPT FPOOL1:GRASSI.GOODIES.FOOD
Grantee R W
EDWARDS X -
GRASSI  X X
SMITH   X X
STONE   X -

1= Help      2= Refresh 3= Return  4= S(Grantee) 5= Sort(W) 6=
7= Backward 8= Forward 9=          10=           11=          12=
=====>
X E D I T 1 File

```

Figure 23. Sample AUTHLIST Screen

Messages and Return Codes

DMSWAU651E AUTHLIST must be issued from FILELIST or DIRLIST
[RC=40]

DMSWAU653E Error executing QUERY AUTHORITY, rc=nn [RC=nn]

DMSWAU1132E Invalid number of operands [RC=24]

DMSWAU1230E AUTHLIST is invalid for minidisk file [RC=00]

DMSWAU1233E Invalid use of REFRESH option [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

<u>Reason</u>	<u>Page</u>
Errors in command syntax	811

DISCARD

DISCARD can be used only in the FILELIST, MACLIST, DIRLIST, RDRLIST, and PEEK command environments.

Use the DISCARD command to erase a file (or directory, or MACLIB member) that is displayed in the list. DISCARD can either be typed in the command area of the line that describes the file you want discarded, or it can be entered from the command line (at the bottom of the screen).

Format

DISCARD	[<i>objectid</i>]
----------------	---------------------

Operands*objectid*

identifies the file (or directory, or MACLIB member) to be discarded. The *objectid* is only necessary when DISCARD is typed on the command line. If DISCARD is typed on the line listing the file (or directory, or MACLIB member), the *objectid* is appended automatically.

The syntax of *objectid* depends on the command environment from which you enter DISCARD.

In the FILELIST environment, *objectid* can be:

fn ft fm

is the file identifier of the file to be erased.

fn ft dirid

is the file identifier of a file in an SFS directory to be erased.

dirid

is the name of the directory to be erased. The directory must be empty of all files and subdirectories.

In the DIRLIST environment, *objectid* is:

dirid

is the name of the directory to be erased. The directory must be empty of all files and subdirectories.

In the MACLIST environment, *objectid* is:

libname libtype libmode (MEMBER membername

is the name of the member to be erased.

In the PEEK environment, do not enter any *objectid* since only one reader file is displayed.

In the RDRLIST environment, *objectid* is:

spoolid

is the spoolid of the reader file to be purged.

Usage Notes

1. For files and SFS directories, DISCARD issues the ERASE command. See the ERASE command for the required authorities to erase a directory or a file in a directory.
2. In most cases, to discard a file in an SFS directory, you must own the directory and have the directory accessed read/write. To erase a file in a directory that you do not own, you must specify the *dirid* on DISCARD. For example, to discard a directory from DIRLIST on the line where the directory is displayed, type:

```
discard /ntd
```

This means discard *fn ft dirid* displayed on the same line.

Messages and Return Codes

DMSWDC651E DISCARD must be issued from FILELIST, RDRLIST, MACLIST, DIRLIST, or PEEK [RC=40]

DMSWDC653E Error executing *command*, rc=*nn*[RC=40]

DMSWDC752E Unable to delete member *membername* from *fn ft fm* [RC=88]

DMSWDC1162E Directory *dirname* is not empty [RC=40]

DMSWDC8348I Directory *dirname* has been discarded

Additional system messages may be issued by this command. The reasons for these messages and the page that lists them are:

Reason	Page
Errors in command syntax	811
Errors in the Shared File System	813

EXECUTE

EXECUTE can be used only in the CSLLIST, DIRLIST, FILELIST, MACLIST, and RDRLIST command environments. Use EXECUTE (an XEDIT macro) to issue commands (or execs) that make use of lists displayed by these commands.

EXECUTE may be used in two ways. First, on a display terminal, the command(s) to be executed can be typed directly on the screen and "EXECUTE" entered either on the command line or from a PF key or by pressing the ENTER key. Second, the command to be executed can be typed in the command line at the bottom of the screen, following "EXECUTE" (as one of its operands). The command is then executed against one or more files in the list, beginning with the current line of the list.

Format

EXECUTE	[Cursor] [<i>command</i>] [<i>lines</i>]
----------------	--

Operands**Cursor**

means that a command is to be executed against the file (member, directory, or CSL routine) that contains the cursor. The command can either be typed on the line that describes the file, or it can be typed as an operand of EXECUTE. The CURSOR operand is valid only on display terminals and is particularly useful when assigned to a PF key. For example, if EXECUTE CURSOR XEDIT is assigned to a PF key, you can place the cursor on the line describing the file you want to edit and then press the PF key.

lines

is the number of lines in the list the command is to be executed for, starting with the file described on the current line of the list. If a command is specified, the default is one (1). You can specify an asterisk (*), which means "execute this command on all lines, from the current line to the end of the list."

command

is a CMS or CP command (or any program or exec) that makes use of files in the list. You can either type out the command operands, or you can use the symbols.

Usage Notes

1. When a command is typed on the screen, EXECUTE rebuilds the line and compares it with the line displayed on the screen. The line is scanned from right to left, and the first character that is different signals the end of the command. Therefore, if the information has been changed (as the result of a previous command), but this information has not yet been updated (by pressing PF2 to refresh the screen), EXECUTE will incorrectly interpret the information on the screen. An example follows.

Sample FILELIST list:

Cmd	Filename	Filetype	Fm	Format	Lrecl	Records	Blocks	Date	Time
	CMS	EXEC	A1	F	80	268	21	1/11/82	13:44:19
	TEST	LIST	A1	F	80	22	2	1/11/82	13:19:29

.....

Issue COPYFILE command:

Cmd	Filename	Filetype	Fm	Format	Lrecl	Records	Blocks	Date	Time
copyfile / test list a		(APPEND			80	268	21	1/11/82	13:44:19
	TEST	LIST	A1	F	80	22	2	1/11/82	13:19:29

.....

After pressing the ENTER key only the line with the command is refreshed:

Cmd	Filename	Filetype	Fm	Format	Lrecl	Records	Blocks	Date	Time
*	CMS	EXEC	A1	F	80	268	21	1/11/82	13:44:19
	TEST	LIST	A1	F	80	22	2	1/11/82	13:19:29

.....

Pressing PF2 updates the other files in the list:

Cmd	Filename	Filetype	Fm	Format	Lrecl	Records	Blocks	Date	Time
	TEST	LIST	A1	F	80	290	23	1/11/82	13:46:38
	CMS	EXEC	A1	F	80	268	21	1/11/82	13:44:19

.....

2. Entering Commands on the Command Line

Another way to issue commands that make use of the lines displayed is to move the current line to the first (or only) line you want the command to use, and then to issue EXECUTE (in the form "EXECUTE lines command") from the XEDIT command line. This method may be used on both display and typewriter terminals.

For example:

First move the current line (by using XEDIT subcommands like UP or DOWN) to the first line you want to use in the command. On a full screen display, the current line is the first file on the screen. Then (in the XEDIT command line) you type:

```
execute n command
```

where *n* is the number of lines the command is executed for, starting with the current line, and *command* is the name of the command.

Note: You can use XEDIT synonyms or macros to make issuing common commands easier. For example, you might want to set up a command "EX" to be a synonym for "EXECUTE 1 XEDIT."

3. Using Symbols as Part of a Command

Symbols can be used to represent operands in the command to be executed. They can be used in the commands typed on the screen, or as part of the command in EXECUTE (on the command line). The symbols are different in each command environment; see the usage notes of each of the commands that use EXECUTE for a list of the symbols.

EXECUTE

Messages and Return Codes

- DMSWEX526E Option CURSOR valid in display mode only [RC=3]
- DMSWEX543E Invalid number: *number* [RC=5]
- DMSWEX561E Cursor is not on a valid data field [RC=1 or 3]
- DMSWEX651E EXECUTE must be issued from RDRLIST, FILELIST, DIRLIST, CSLLIST, or MACLIST [RC=40]
- DMSWEX654E Invalid symbol *symbol*; {/0 must be specified alone|invalid character *char* following / symbol} [RC=24]

On a typewriter terminal only:

Executing: command
+++E(nn)+++

Chapter 5. HELP and HELPCONV Format Words

This part describes the formats, operands, and defaults of the HELP facility format words. In each of the format word descriptions, the default values are those that are implied when you enter a format word with no operands or parameters. For example, the default operand of the .FO (FORMAT MODE) format word is “on”. Therefore, the format lines

```
.fo
.fo on
```

are equivalent, and in the format box of the .FO format word, the “on” operand is underscored.

HELP format words are used in HELP files when you want HELP to do output formatting when the file is processed. Table 25 gives a summary of the HELP facility format words. All format words, except .CM, .CS, and .MT, are used expressly with the HELPCONV command. The HELPCONV module converts the specified file into a formatted HELP file, leaving the .CM, .CS, and .MT control words in the file. The output file has the filetype changed from ‘HELPxxxx’ to ‘\$HLPxxxx’.

FORMAT WORD	OPERAND FORMAT	FUNCTION	BREAK	DEFAULT VALUE
.BX (BOX)	V1 V2...Vn OFF	Draws horizontal and vertical lines around subsequent output text in blank columns.	Yes	Draws a horizontal line.
.CM (COMMENT)	Comments	Places comments in a file for future reference.	Yes	
.CS (CONDITIONAL SECTION)	n/keyword ON/OFF	Allows conditional inclusion of input in the formatted output.	Yes	
.FO (FORMAT MODE)	ON/OFF	Causes concatenation of input lines, and left and right justification of output.	Yes	On
.IL (INDENT LINE)	n +n -n	Indents only the next line the specified number of spaces.	Yes	0
.IN (INDENT)	n +n -n	Specifies the number of spaces subsequent text is to be indented.	Yes	0
.MT (MENU TYPE)	component	Correlates a component to a menu file when the component is not to be derived from the filename. For files other than menu files, .MT is seen as a comment.	Yes	
.OF (OFFSET)	n +n -n	Provides a technique for indenting all but the first line of a section.	Yes	0
.SP (SPACE)	n	Specifies the number of blank lines to be inserted before the next output line.	Yes	1
.TR (TRANSLATE)	s t	Specifies the final output representation of any input character.	No	

.BX (BOX)

The BOX format word defines and initializes a horizontal rule for output and defines vertical rules for subsequent output lines.

The format of the .BX format word is:

.BX	$\left[\begin{array}{c} v1 \quad v2 \quad [\dots [vn]] \\ \text{OFF} \end{array} \right]$
-----	--

where:

v1-vn

are the positions at which you want to place vertical rules in output text. This format of the format word initializes the box and draws a horizontal line with vertical descenders at the columns indicated. Subsequently entering the .BX format word with no operands causes HELPCONV to create a horizontal line with vertical bars at the columns indicated. The maximum value that may be entered for operands v1-vn is 239.

Off

causes HELP to finish drawing the box by printing a horizontal line with vertical ascenders at the columns specified in a previous .BX format word.

1. The .BX format word describes an overlay structure for subsequent text that is processed by HELPCONV. After the '.BX v1 v2 ...' line is processed, HELPCONV continues processing output lines as usual. However, before a line is printed, HELPCONV places vertical bars in the columns indicated by v1, v2, and so on, unless a column is already occupied by a data character. In this case, HELPCONV does not place a vertical bar in the column.
2. The .BX control word causes a break in the text.
3. The terminal output characters for boxes are formed with dashes (-), vertical bars (|), and plus signs (+).
4. You can specify a .BX format word with different columns while a box is being drawn. When this happens, HELPCONV puts in vertical ascenders for all the old columns and vertical descenders for all the new columns. The vertical rules then appear in all subsequent output lines in the new columns designated.
5. The column specification for the .BX format word uses a different rule than is used elsewhere in HELPCONV. In some control words the numbers in the format word represent not columns but displacements. For example the HELPCONV format word .IN 5 means that a blank character should be expanded to enough blanks to fill up *through* column 5; the next word starts in column 6. In the .BX control word, .BX 5 means to put vertical rules *in* column 5. Thus, you can use the same numbers for a .IN control word as for a .BX control word, and the vertical bar will appear in the column immediately preceding the first word on that line.

Example

Consider the HELP file called 'MARYHADA' that looks like this:

```
.fo off  
.bx 1 43  
.in 5  
Mary had a little lamb,  
Whose fleece was white as snow,  
And everywhere that Mary went,  
The lamb was sure to go.  
.bx off
```

This file, when processed by HELPCONV, creates the following output:

```
Mary had a little lamb,  
Whose fleece was white as snow,  
And everywhere that Mary went,  
The lamb was sure to go.
```

.CM (COMMENT)

Use the COMMENT format word to place comments within a HELP file.

The format of the .CM format word is:

.CM	<i>comments</i>
------------	-----------------

where:

comments

may be anything; this input line is not used in formatting the output.

Usage Notes

1. The .CM format word enables you to store comments in the HELP files for future reference. Comment lines are retained in the formatted file and do not appear when the HELP file is displayed.
2. You can use comments to store unique identifications to be used to locate a specific region of the file during editing.
3. This format word acts as a break.

Example

.CM Remember to change the date.

The line above is seen only when editing the HELP file, and it reminds you to change the date used in the text.

.CS (CONDITIONAL SECTION)

The **CONDITIONAL SECTION** format word identifies to **HELP** the section of the input file that is to be displayed based on the specified **HELP** command option.

The format of the **.CS** format word is:

.CS	[<i>n</i> <i>keyword</i>]	[ON OFF]
------------	--------------------------------	-----------------------------

where:

n

is a number from 0 to 7, which corresponds to the conditional section of the **HELP** file.

keyword

is the name of the conditional section. It may be:

BRIEF
DESCRIPT
FORMAT
PARMS
OPTIONS
NOTES
ERRORS
RELATED

These names correspond to the conditional section code number.

on

marks the beginning of conditional section *n*.

off

marks the end of conditional section *n*.

Usage Notes

1. The **.CS** format word enables you to identify the specific sections of the input file that are directly associated with the **HELP** facility command "options."

You can then specify which section(s) of the **HELP** file are to be displayed by using the **HELP** command options: **BRIEF**, **DESCRIPT**, **FORMAT**, **PARMS**, **OPTIONS**, **NOTES**, **ERRORS**, and **RELATED**.

If you choose to implement any **HELP** command files using the **HELP** command options, either the format word '**.CS n ON**' or the '**.CS keyword ON**' is required in the file. You must use the following form:

HELP and HELPCONV Format Words

Top of file
.CS 0 on (or .CS BRIEF on)
 (Text for BRIEF option)
.CS 0 off (or .CS BRIEF off)
.CS 1 on (or .CS DESCRIPT on)
 (Text for DESCRIPT option)
.CS 1 off (or .CS DESCRIPT off)
.CS 2 on (or .CS FORMAT on)
 (Text for FORMAT option)
.CS 2 off (or .CS FORMAT off)
.CS 3 on (or .CS PARMS on)
 (Text for PARMS option)
.CS 3 off (or .CS PARMS off)
.CS 4 on (or .CS OPTIONS on)
 (Text for OPTIONS option)
.CS 4 off (or .CS OPTIONS off)
.CS 5 on (or .CS NOTES on)
 (Text for NOTES option)
.CS 5 off (or .CS NOTES off)
.CS 6 on (or .CS ERRORS on)
 (Text for ERRORS option)
.CS 6 off (or .CS ERRORS off)
.CS 7 on (or .CS RELATED on)
 (Text for RELATED option)
.CS 7 off (or .CS RELATED off)
End of file

2. This format word acts as a break.
3. If blank lines or portions of the file are written between the conditional sections (.CS sections), these lines are considered uncontrolled data and will be displayed with the **DETAIL** information.

Notes:

- Few **RELATED HELP** files exist, but you can use this option to customize your own **HELP** files.
- Abbreviations of conditional keywords are not allowed.

.FO (FORMAT MODE)

Use the FORMAT MODE format word to cancel or restore concatenation of input lines and right-justification of output lines.

The format of the .FO format word is:

.FO	[<u>ON</u> OFF]
-----	----------------------

where:

ON

restores default HELPCONV formatting, including both justification and concatenation of lines. If you use the .FO format word with no operands, ON is assumed.

OFF

cancels concatenation of input lines and justification of output lines. Subsequent text is printed "as is."

Usage Notes

1. When format mode is in effect, lines are formed by shifting words to or from the next line (concatenation) and padding with extra blanks to produce an aligned right margin (justification).
2. This format word acts as a break.
3. When format mode is in effect, a line without any blanks that exceeds the current line length is extended into the right margin. If a line is processed so that only one word fits on the line, the word is left-justified.
4. If *no* formatting is done by HELPCONV, HELP description files *must* contain a ".fo off" format word as the first line of the file.
5. HELP MENU/TASK files *must* contain a ".fo off" format word as the first line of the file if the HELPCONV command is to be used. For files with RELATED sections, the ".fo off" format line *must* precede the RELATED section if the HELPCONV command is to be used.
6. If the HELP files are used on releases of CMS prior to VM/SP Release 4, add a ".fo off" format word as the first line in the file. You should also change any ".CS" lines that contain keywords or section numbers not supported in prior releases to comment lines (.CM).

Examples

1. .FO OFF

Justification and concatenation are completed for the preceding line or lines, but the following lines are typed exactly as they appear in the file.

2. .FO

HELP and HELPCONV Format Words

Justification and formatting are resumed with the next input line. Output from this point on in the file is padded to produce an aligned right margin on the output page.

.IL (INDENT LINE)

Use the INDENT LINE format word to indent the *next line only* a specified number of characters.

The format of the .IL format word is:

.IL	$\begin{bmatrix} n \\ +n \\ -n \end{bmatrix}$
------------	---

where:

n

specifies the number of character spaces to shift the next line from the current margin. +n specifies that text is shifted to the right, and -n shifts text to the left.

Usage Notes

1. The .IL format word provides a way to indent the next output line. The line is shifted to the right or the left of the current margin (which includes any indent or offset values in effect).
2. This format word acts as a break.
3. The .IL format word is useful for beginning new paragraphs.
4. When successive .IL format words are encountered without intervening text, or when you specify positive or negative increments for .IL format words entered without intervening text, the indent amount is modified to reflect the last .IL encountered; that is, the increments are added together. Thus the lines:

```
.il 4
.il +6
```

result in the next line being indented 10 spaces.

5. When you use the .IL format word with a negative value (undenting), an error message is generated if the resulting amount would cause a shift to the left of character position one.

.IN (INDENT)

Use the INDENT format word to change the left margin displacement of HELP output.

The format of the .IN format word is:

.IN	$\left[\begin{array}{c} n \\ +n \\ -n \\ \underline{0} \end{array} \right]$
------------	--

where:

n specifies the number of spaces to be indented. If omitted, 0 is assumed, and indentation reverts to the left margin. If you use +n or -n, the current left margin increases or decreases by the amount specified.

Usage Notes

1. The .IN format word resets the current left margin. This indentation remains in effect for all following lines until another .IN format word is encountered. “.in 0” cancels the indentation, and output continues at the original left margin setting.
2. The value of n represents the number of blank spaces left before text margins. Thus, “.in 5” sets the left margin at column 6, leaving 5 blank spaces at the left.
3. This format word acts as a break.
4. The .IN format word cancels any .OF (OFFSET) setting. The “.of 0” request cancels the current offset, but leaves the left margin specified by the .IN format word unchanged.

.MT (MENU TYPE)

Use the MENU TYPE format word specify the component of a menu. The format of the .MT format word is:

.MT	<i>component</i>
-----	------------------

where:

component

is the component used by the menu.

Usage Notes

1. The .MT format word is used to assist in the creation of menus that are a subset of another menu.
2. This format word acts as a break.

Example

A menu that contains a list of REXX functions might be called FUNCTION HELPMENU. In this case, the HELP files for the individual functions can only be located if they are duplicated under the filetype "HELPFUNC." The .MT control word defines a component id to override that derived from the filename. The FUNCTION menu could be:

```
.MT REXX
```

to specify that the menu contains a list of REXX commands and thus will be found under the filetype "HELPREXX".

.OF (OFFSET)

Use the OFFSET format word to indent all but the first line of a block of text.

The format of the .OF format word is:

.OF	$\left[\begin{array}{c} n \\ +n \\ -n \\ \underline{0} \end{array} \right]$
-----	--

where:

n

specifies the number of spaces to be indented after the next line is formatted. If omitted, 0 is assumed, and indentation reverts to the original margin setting. If you use +n or -n, the current offset value increases or decreases the specified amount, and a new offset is started.

Usage Notes

1. The .OF format word does not take effect until after the next line is formatted. The indentation remains in effect until a .IN (INDENT) format word or another OFFSET control word is encountered.

You can use the .OF format word within a section that is also indented with the .IN format word. Note that .IN settings take precedence over .OF, however, and any .IN request causes a previous offset to be cleared.

If you want to start a new section with the same offset as the previous section, you need only repeat the .OF n request.

2. This format word acts as a break.
3. You can use the .IL (INDENT LINE) format word to shift only the next line to the left or right of the current margin.

Example

1. Starting an offset:

.of 10

The line immediately following the .OF format word is printed at the current left margin. All lines thereafter (until the next indent or offset request) are indented ten spaces from the current margin setting. These two examples were processed with OFFSET control words in the positions shown.

2. Ending an offset:

.of

The effect of any previous .OF request is canceled, and all output after the next line continues at the current left margin setting.

.SP (SPACE LINES)

Use the SPACE LINES format word when you want blank lines to appear between text lines of output.

The format of the .SP format word is:

.SP	$\left[\begin{array}{c} n \\ \underline{1} \end{array} \right]$
------------	--

where:

n

specifies the number of blank lines to be inserted in the output. If omitted, 1 is assumed.

.TR (TRANSLATE CHARACTER)

The TRANSLATE CHARACTER format word allows you to specify the output representation of each character in the source text. For example, you could specify that all exclamation points in the file appear as blanks in the output.

The format of the .TR format word is:

.TR	[<i>s t</i>]
------------	----------------

where:

s

is a source character under consideration. It may be a single character or a two-character hexadecimal code.

t

is the intended output representation of the source character. It may be a single character or a two-character hexadecimal code.

Usage Notes

1. After formatting of an input source line has been completed and immediately before actual output, each character of the output line may be translated to a different output code.
2. Since format words are only processed internally, they are never translated in the file.
3. Translate character specifications remain in effect until explicitly respecified.
4. A .TR format word with no operands causes the translation table to be reinitialized and all previously specified translations to be reset.
5. The .TR format word does not cause a break. If you have a section of text that has translation characters in effect, followed by a .TR to reset the translations, the last line of the text may not yet have been printed. In this case, that last line is not translated.

Example

```
.tr 40 ?
```

This causes all blanks in the file to be typed as question marks (?) on output.

Chapter 6. System Messages

This chapter contains system messages that you might receive when you issue a command that:

- contains syntax errors
- interacts with the Shared File System
- uses or changes files.

The messages and their return codes are listed below.

Command Syntax Error Messages

For some CMS commands, you may receive messages for syntax errors, that is, for specifying the command format incorrectly. These error message are prefixed by DMSPCL, having a return code of 24. Check the format of the command, make the necessary correction, and enter the command again.

For example, if you issue the WRITE VSCREEN command as WRITE, you receive the message:

DMSPCL384E Missing modifier keyword(s)

Check the format of the WRITE VSCREEN command and enter it again, using the correct command name and operands.

Number	Message
384E	Missing modifier keyword(s)
385E	Invalid modifier keyword: <i>keyword</i>
386E	Missing operand(s)
387E	Missing value for <i>operand</i> operand
387E	Missing alphanumeric string for <i>operand</i> operand
387E	Missing application identifier for <i>operand</i> operand
387E	Missing character for <i>operand</i> operand
387E	Missing device address for <i>operand</i> operand
387E	Missing filename for <i>operand</i> operand
387E	Missing filetype for <i>operand</i> operand
387E	Missing execname for <i>operand</i> operand
387E	Missing exectype for <i>operand</i> operand
387E	Missing filemode for <i>operand</i> operand
387E	Missing hexadecimal number for <i>operand</i> operand
387E	Missing integer for <i>operand</i> operand
387E	Missing negative integer for <i>operand</i> operand
387E	Missing number for <i>operand</i> operand
387E	Missing positive integer for <i>operand</i> operand
387E	Missing mode for <i>operand</i> operand
387E	Missing character string for <i>operand</i> operand
387E	Missing directory id for <i>operand</i>
387E	Missing namedef for <i>operand</i>
387E	Missing filepoolid for <i>operand</i>
387E	Missing filemode or directory id for <i>operand</i>
388E	Invalid keyword: <i>keyword</i>
389E	Invalid operand: <i>operand</i>
389E	Invalid alphanumeric string: <i>string</i>
389E	Invalid application identifier: <i>string</i>
389E	Invalid character: <i>character</i>

Number	Message
389E	Invalid device address: <i>address</i>
389E	Invalid filename: <i>filename</i>
389E	Invalid filetype: <i>filetype</i>
389E	Invalid execname: <i>execname</i>
389E	Invalid exectype: <i>exectype</i>
389E	Invalid filemode: <i>filemode</i>
389E	Invalid hexadecimal number: <i>number</i>
389E	Invalid integer: <i>integer</i>
389E	Invalid negative integer: <i>integer</i>
389E	Invalid number: <i>integer</i>
389E	Invalid positive integer: <i>integer</i>
389E	Invalid mode: <i>mode</i>
389E	Invalid character string: <i>string</i>
389E	Invalid directory id: <i>dirid</i>
389E	Invalid namedef: <i>namedef</i>
389E	Invalid filepoolid: <i>filepoolid</i>
389E	Invalid filemode or directory id: <i>value</i>
390E	Invalid value <i>value</i> for <i>operand</i> operand
390E	Invalid alphanumeric string <i>string</i> for <i>operand</i> operand
390E	Invalid application identifier <i>applid</i> for <i>operand</i> operand
390E	Invalid character <i>character</i> for <i>operand</i> operand
390E	Invalid device address <i>address</i> for <i>operand</i> operand
390E	Invalid filename <i>filename</i> for <i>operand</i> operand
390E	Invalid filetype <i>filetype</i> for <i>operand</i> operand
390E	Invalid execname <i>execname</i> for <i>operand</i> operand
390E	Invalid exectype <i>exectype</i> for <i>operand</i> operand
390E	Invalid filemode <i>filemode</i> for <i>operand</i> operand
390E	Invalid hexadecimal number <i>number</i> for <i>operand</i> operand
390E	Invalid integer <i>integer</i> for <i>operand</i> operand
390E	Invalid negative integer <i>integer</i> for <i>operand</i> operand
390E	Invalid number <i>number</i> for <i>operand</i> operand
390E	Invalid positive integer <i>integer</i> for <i>operand</i> operand
390E	Invalid mode <i>mode</i> for <i>operand</i> operand
390E	Invalid character string <i>string</i> for <i>operand</i> operand
390E	Invalid directory id <i>dirid</i> for <i>operand</i> operand
390E	Invalid namedef <i>namedef</i> for <i>operand</i> operand
390E	Invalid filepoolid <i>filepoolid</i> for <i>operand</i> operand
390E	Invalid filemode or directory id <i>value</i> for <i>operand</i> operand
391E	Unexpected operand(s): <i>operand</i>
393E	Missing value for <i>option</i> option
393E	Missing alphanumeric string for <i>option</i> option
393E	Missing application identifier for <i>option</i> option
393E	Missing character for <i>option</i> option
393E	Missing device address for <i>option</i> option
393E	Missing filename for <i>option</i> option
393E	Missing filetype for <i>option</i> option
393E	Missing execname for <i>option</i> option
393E	Missing exectype for <i>option</i> option
393E	Missing filemode for <i>option</i> option
393E	Missing hexadecimal number for <i>option</i> option
393E	Missing integer for <i>option</i> option
393E	Missing negative integer for <i>option</i> option
393E	Missing number integer for <i>option</i> option
393E	Missing positive integer for <i>option</i> option
393E	Missing mode for <i>option</i> option
393E	Missing character string for <i>option</i> option
393E	Missing directory id for <i>option</i> option
393E	Missing namedef for <i>option</i> option
393E	Missing filepoolid for <i>option</i> option
393E	Missing filemode or directory id for <i>option</i> option
394E	Invalid option: <i>option</i>
395E	Invalid value <i>value</i> for <i>option</i> option
395E	Invalid alphanumeric string <i>string</i> for <i>option</i> option
395E	Invalid application identifier <i>applid</i> for <i>option</i> option

Number	Message
395E	Invalid character <i>character</i> for <i>option</i> option
395E	Invalid device address <i>address</i> for <i>option</i> option
395E	Invalid filename <i>filename</i> for <i>option</i> option
395E	Invalid filetype <i>filetype</i> for <i>option</i> option
395E	Invalid execname <i>execname</i> for <i>option</i> option
395E	Invalid exectype <i>exectype</i> for <i>option</i> option
395E	Invalid filemode <i>filemode</i> for <i>option</i> option
395E	Invalid hexadecimal number <i>number</i> for <i>option</i> option
395E	Invalid integer <i>integer</i> for <i>option</i> option
395E	Invalid number <i>number</i> for <i>option</i> option
395E	Invalid negative integer <i>integer</i> for <i>option</i> option
395E	Invalid positive integer <i>integer</i> for <i>option</i> option
395E	Invalid mode <i>mode</i> for <i>option</i> option
395E	Invalid character string <i>string</i> for <i>option</i> option
395E	Invalid directory id <i>dirid</i> for <i>option</i> option
395E	Invalid namedef <i>namedef</i> for <i>option</i> option
395E	Invalid filepoolid <i>filepoolid</i> for <i>option</i> option
395E	Invalid filemode or directory id <i>value</i> for <i>option</i> option

Shared File System (SFS) Error Messages

If you enter a command that interacts with the Shared File System, you may receive an error message with the prefix of DMSSDM. For a detailed description of a message and the suggested action to resolve the error, see the *VM/SP System Messages and Codes*.

Number	Message
109S	Virtual storage capacity exceeded [RC = 31 104]
1137E	Object is locked [RC = 31 70]
1137E	Object is locked; deadlock detected [RC = 31 70]
1138E	Filesharing conflict [RC = 31 70]
1139E	You are not authorized to issue this command [RC = 31 76]
1141W	User filespace threshold still exceeded for file pool <i>filepoolid</i> [RC = 4]
1142E	Error reading system catalog for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1142E	Error writing system catalog for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1142E	Error in file access function for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1142E	Error in locking function for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1142E	Error in query function for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1142E	Error in storage management for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1142E	Error <i>nn</i> for file pool <i>filepoolid</i> ; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> [RC = 31 104]
1143E	Inconsistent catalogs in file pool <i>filepoolid</i> [; error codes <i>nn</i> and <i>nn</i> ; Detecting module <i>moduleid</i> RC = 31 104]
1145E	Further communication with file pools is impossible [RC = 31 104]
1146E	Deadlock <i>code</i> encountered for file pool <i>filepoolid</i> [RC = 31 104]
1146E	File pool limit <i>code</i> encountered for file pool <i>filepoolid</i> [RC = 31 104]
1146E	I/O error <i>code</i> encountered for file pool <i>filepoolid</i> [RC = 31 104]
1146E	File pool catalog space error <i>code</i> encountered for file pool <i>filepoolid</i> [RC = 31 104]
1147E	Storage management error trying to free storage [RC = 31 104]
1147E	Storage management error trying to get storage [RC = 31 104]
1148E	APPC/VM IDENTIFY error [RC = 31 55]
1148E	APPC/VM error [RC = 31 55]
1149E	Error occurred in user exit routine [RC = 31 40]
1150E	Error occurred while calling user exit routine [RC = 31 40]
1151E	File pool <i>filepoolid</i> is unavailable [RC = 31 55]
1153E	File pool <i>filepoolid</i> is unavailable or unknown [RC = 31 99]
1155E	CSL routine <i>cslname</i> is not loaded [RC = 40]
1155E	CSL routine <i>cslname</i> has been dropped [RC = 40]

Number	Message
1156S	Supervisor error 1: return code <i>nm</i> [RC = 31 104]
1156S	Supervisor error 2: return code <i>nm</i> , reason code <i>nm</i> [RC = 31 104]
1157E	Work unit already active when atomic request is issued for work unit <i>workunitid</i> [RC = 31 70]
1158E	Attempt to make uncommitted updates to more than one file pool on work unit <i>workunitid</i> [RC = 31 70]
1159E	User has files or directories open when a COMMIT is requested on work unit <i>workunitid</i> [RC = 31 70]
1164E	Command <i>commandname</i> failed; storage group being restored [RC = 28 31]
1174E	You have tried to establish more APPC/VM connections than is allowed for your userid [RC = 31 55]
1174E	Your attempt exceeds the number of APPC/VM connections allowed for file pool <i>filepoolid</i> [RC = 31 55]
1176E	Virtual storage capacity exceeded for file pool <i>filepoolid</i> [RC = 31 99]
1179E	<i>filepoolid</i> is a remote file pool that was started for local use only [RC = 31 99]
1212E	You have opened a file pool catalog for WRITE on work unit <i>workunitid</i> for file pool <i>filepoolid</i> [RC = 31 40]
1223E	There is no default file pool currently defined [RC = 40]
1240E	You are not authorized to connect to file pool <i>filepoolid</i> [RC = 31 76]
1254E	An attempt to commit exceeded the number of 4K blocks allowed for the user in file pool <i>filepoolid</i> [RC = 31 40]
1259E	File pool <i>filepoolid</i> has run out of physical space in the storage group [RC = 31]
1265E	A request was in process when you requested a COMMIT for work unit <i>workunitid</i> [RC = 31 70]
1265E	A COMMIT was in process when you issued a request for work unit <i>workunitid</i> [RC = 31 70]
1311E	Object already exists [RC = 28 31 70]

File Error Messages

If you enter a command that uses or changes files, you may receive one of the following error message. For a detailed description of a message and the suggested action to resolve the error, see the *VM/SP System Messages and Codes*.

Message ID	Message
DMSOPN002E	File <i>fn ft fm</i> not found [RC = 28]
DMSOPN024E	File <i>fn ft fm</i> already exists [RC = 35]
DMSOPN030E	File <i>fn ft fm</i> already active [RC = 37]
DMSOPN037E	Filemode <i>fm</i> is accessed as read/only [RC = 12]
DMSOPN048E	Invalid filemode <i>fm</i> [RC = 24]
DMSOPN062E	Invalid character <i>char</i> in fileid <i>fn ft [fm]</i> [RC = 20]
DMSOPN062E	SO and SI are invalid fileid characters [RC = 20]
DMSOPN069E	Filemode <i>mode</i> not accessed [RC = 36]
DMSOPN109S	Insufficient free storage available [RC = 25]
DMSOPN170S	Disk <i>mode(vdev)</i> has maximum number of files [RC = 10]
DMSOPN1138E	File sharing conflict involving file <i>fn</i> [RC = 70]
DMSOPN1258E	You are not authorized to write to file <i>fn ft fm</i> [RC = 28]
DMSOPN1260E	Invalid OPENTYP <i>xx</i> specified in FSCB for file <i>fn ft fm</i> [RC = 33]
DMSOPN1261E	Invalid CACHE specified in FSCB for file <i>fn ft fm</i> [RC = 34]
DMSOPN1262S	Error <i>nm</i> opening file <i>fn ft fm</i> [RC = 3 11 80 81 82 83 84]
DMSSTT048E	Invalid filemode <i>mode</i> [RC = 24]
DMSSTT229E	Unsupported OS dataset, error [80 81 82 83]
DMSSTT253E	File <i>fn ft fm</i> cannot be handled with supplied parameter list [RC = 88]

Appendix A. VSE/VSAM Functions Not Supported in CMS

Refer to the publication *Using VSE/VSAM Commands and Macros* for a description of Access Method Services functions available under VSE, and, therefore, under CMS. This knowledge of Access Method Services is assumed throughout this publication.

All of VSE/VSAM is supported by CMS, except for the following:

- Non-VSAM data sets with data formats that are not supported by CMS/DOS (for example, BDAM and ISAM files are not supported).
- The SHAREOPTIONS operand is not supported for cross system or cross partition sharing in CMS/DOS (that is, DASD sharing is not supported).
- Space Management for SAM Feature
- Backup/Restore Feature

If an AMSERV input file to VSE/VSAM Access Method Services contains the control statement "DELETE" with "IGNORERROR," the PRINT option on the AMSERV command must be used to send the output to the virtual printer.

Appendix B. OS/VS Access Method Services and VSAM Functions Not Supported

In CMS, an OS user is defined as a user that has *not* issued the command:

```
SET DOS ON (VSAM)
```

OS users can use all of the Access Method Services functions that are supported by VSE/VSAM, with the following exceptions:

- Non-VSAM data sets with data formats that are not supported by CMS/DOS (for example, BDAM and ISAM files are not supported).
- The SHAREOPTIONS operand is not supported for cross system or cross partition sharing in CMS/DOS (that is, DASD sharing is not supported).
- Do not use the AUTHORIZATION (entrypoint) operand in the DEFINE and ALTER commands unless your own authorization routine exists on the DOS core image library, the private core image library, or in a CMS DOSLIB file. In addition, results are unpredictable if your authorization routine issues an OS SVC instruction.
- The OS Access Method Services GRAPHICS TABLE options and the TEST option of the PARM command are not supported.
- The filename in the FILE (filename) operands is limited to seven characters. If an eighth character is specified, it is ignored.
- The OS access method services CNVTCAT and CHKLIST commands are not supported in VSE/VSAM access method services. In addition, all OS access method services commands that support the 3850 Mass Storage System are not supported in DOS/VS access method services.
- Table 26 is a list of OS operands, by control statement, that are not supported by the CMS interface to VSE/VSAM Access Method Services.

If any of the unsupported operands or commands in Table 26 are specified, the AMSERV command terminates and displays an appropriate error message.

When you use the PRINT, EXPORT, IMPORT, IMPORTRA, EXPORTRA, and REPRO control statements for sequential access method (SAM) data sets, you must specify the ENVIRONMENT operand with the required DOS options (that is, PRIME DATA DEVICE, BLOCKSIZE, RECORDSIZE, or RECORDFORMAT). You must have previously issued a DLBL for the SAM file.

AMSERV can write SAM data sets only to a CMS disk or directory, but can read them from DOS, OS, or CMS disks or directories.

Table 26 (Page 1 of 2). OS Access Method Service Operands NOT Supported in CMS	
OS ACCESS METHOD SERVICES CONTROL STATEMENT	OPERANDS NOT SUPPORTED IN CMS
ALTER	EMPTY/NOEMPTY SCRATCH/NOSCRATCH DESTAGWAIT/NODESTAGWAIT STAGE/BIND/CYLINDERFAULT

Table 26 (Page 2 of 2). OS Access Method Service Operands NOT Supported in CMS	
OS ACCESS METHOD SERVICES CONTROL STATEMENT	OPERANDS NOT SUPPORTED IN CMS
DEFINE	ALIAS EMPTY/NOEMPTY GENERATIONDATAGROUP PAGESPACE SCRATCH/NOSCRATCH DESTAGEWAIT/NODESTAGEWAIT STAGE/BIND/CYLINDERFAULT TO/FOR/OWNER ¹¹
DELETE	ALIAS GERATIONDATAGROUP PAGESPACE
EXPORT	OUTDATASET
IMPORT	INDATASET OUTDATASET IMPORTA
LISTCAT	ALIAS GENERATIONDATAGROUP LEVEL OUTFILE ¹² PAGESPACE
PRINT	INDATASET OUTFILE ¹²
REPRO	INDATASET OUTDATASET

OS users can use a subset of OS/VSAM Assembler Language Macros in their assembler language programs. The OS/VSAM Assembler language macros supported for use in CMS are contained in the OSVSAM MACLIB that is distributed with the VM/SP. The macros and options that are *not* supported are shown in Table 27.

¹¹ The TO/FOR/OWNER operands are supported for the access method services interface, but are not supported for the DEFINE NONVSAM control statement.

¹² The OUTFILE operand is supported by the access method services interface, but is not supported for the LISTCAT and PRINT control statements.

Table 27. Options of OS/VSAM Macros Not Supported in CMS

OS/VSAM Macro	Options that are Not Supported
ACB	BSTRNO = number CATALOG = YES NO CRA = SCRA UCRA MACRF = CFX NFX, DDN DSN, ICI NCI, NIS SIS, LSR GSR, DFR
EXLST GENCB BLK = ACB	UPAD = address AM = VSAM BSTRNO = number CATALOG = YES NO CRA = SCRA UCRA MACRF = CFX NFX, DDN DSN, ICI NCI, NIS SIS, GSR
GENCB BLK = EXLST GENCB BLK = RPL	AM = VSAM MSGAREA = address MSGLEN = number OPTCD = NWAITX WAITX
MODCB BLK = ACB	AM = VSAM BSTRNO = number CATALOG = YES NO CRA = SCRA UCRA MACRF = CFX NFX, DDN DSN, ICI NCI, NIS SIS, GSR
MODCB BLK = RPL	MSGAREA = address MSGLEN = number OPTCD = NWAITX WAITX
RPL	MSGAREA = address MSGLEN = number OPTCD = NWAITX WAITX
SHOWCB (ACB) SHOWCB (RPL) TESTCB (ACB)	FIELDS = BFRFND, BSTRNO, BUFRDS, ENDRBA, HALCRBA, NUIW, UIW FIELDS = MSGAREA, MSGLEN CATALOG = YES NO CRA = SCRA UCRA MACRF = CFX DDN DSN GSR ICI NCI NFX NIS SIS
TESTCB (RPL)	BSTRNO = number ENDRBA = number MSGAREA = address MSGLEN = number

Summary of Changes

Structural Changes

For VM/SP Release 6, structural changes to this book are:

- Chapter deletion and additions
- Deletion of three appendices

Chapter changes:

- Deleted: DEBUG Subcommands

The CMS command DEBUG no longer places you in the DEBUG subcommand environment. See the DEBUG command description for details.

- Added: CMS Commands for Windowing

This chapter contains windowing and virtual screen commands to create and modify windows and virtual screens, including Border commands to manage your windows.

- Added: Special Commands Used in Command Environments

This chapter contains special CMS commands that you can use from certain command environments such as CSLLIST, DIRLIST, FILELIST, MACLIST, and RDRLIST.

- Added: System Messages

This chapter lists messages that you might receive for errors issuing commands or using CMS files or Shared File System directories.

Appendix changes:

- Deleted: Appendix C. Edit Subcommands and Macros
- Deleted: Appendix D. Edit Reserved Filetype Defaults
- Deleted: Appendix E. CMS EXEC Control Statements

Use the HELP command to see information about Edit subcommands and EXEC control statements.

Technical Changes

Summary of Changes
for SC19-6209-5
for VM/SP Release 6

How to Obtain Prior Editions of This Publication

To obtain editions of this publication that pertain to earlier releases of VM/SP, refer to the *VM/SP Library Guide and Master Index* for the appropriate order number.

New commands for Release 6 of VM/SP

The following commands are new for Release 6.

ALIALIST

Displays alias information in a full-screen environment. You can issue the command from the FILELIST command line or by pressing PF9 from FILELIST (SHARE or FILELIST (SEARCH.

AUHLIST

Displays authority information in a full-screen environment. You can issue the command from the FILELIST command line or by pressing PF6 from from FILELIST (SHARE or FILELIST (SEARCH or from the DIRLIST screen.

CREATE ALIAS

Creates an additional name for a file in a specified directory.

CREATE DIRECTORY

Creates an SFS directory.

CREATE LOCK

Creates an explicit lock on an SFS directory or a file in an SFS directory.

CREATE NAMEDEF

Assigns a temporary name that can be used by a program instead of a file name and file type or a complete directory name.

CSLLIST

Lists information about members of a specified callable services library.

DELETE LOCK

Releases the explicit lock placed on a directory or a file in a directory by the CREATE LOCK command.

DELETE NAMEDEF

Deletes the temporary name given to a file or directory by the CREATE NAMEDEF command.

DIRLIST

Lists directories of a specified directory structure in a full- screen environment.

GRANT AUTHORITY

Authorizes other users to read and/or modify your SFS directories or files within the directories.

LISTDIR

Lists directories in a specified directory structure.

NETDATA

Used from an exec to query, receive or send files to users at a network node or on your system.

PROGMAP

Displays or places on the program stack information on programs currently loaded in storage or in a saved segment.

QUERY

See the following section, Modified Commands for Release 6, for new additions to the Query command.

RELOCATE

Moves a file or directory structure from one directory to another that you own within the same file pool.

REVOKE AUTHORITY

Cancels authorities that you granted to other users for a directory or files in a directory.

RTNDROP

Cancels the binding of a callable services library routine.

RTNLOAD

Searches for, loads, and binds a callable services library routine to a fixed location in storage, and makes it available for invocation.

RTNMAP

Displays information about the callable services library routines that are currently loaded and bound to an address.

RTNSTATE

Obtains the status of one or more specific callable services library routines.

SEGMENT

Manages saved segments

SEGMENT ASSIGN, SEGMENT LOAD, SEGMENT PURGE, SEGMENT RELEASE, and SEGMENT RESERVE

SET

See the following section, Modified Commands for Release 6, for additions to the SET command.

SETKEY

Changes settings for CMS storage keys.

Note: The commands that are authorized for SFS Administrator only are documented in the SFS Administration Guide.

Modified Commands for Release 6**ACCESS**

Addition of *dirid* so that you can access Shared File System (SFS) directories.

CMSSERV

Added two options: CUT and DFT.

DEFAULTS

Options for the following commands have been added to the valid options that you can specify as defaults: CMSSERV, CSLLIST, DIRLIST, DISK LOAD, FILELIST, NETDATA RECEIVE, NETDATA SEND, and READCARD.

DISK

Default options changed to MINPROMPT and NOREPLACE.

ERASE

Addition of *dirid* so that you can erase an SFS directory. New options: STACK, FIFO, LIFO, and FILES, NOFILES.

EXECMAP

New options: SEGMENT and NOSEGMENT

FILELIST

Contains new environments of ALIALIST and AUTHLIST. New options: ALLFILE, AUTHFILE; STATS, SHARE, and SEARCH.

GLOBAL

New option: CSLLIB.

HELP

New HELP on components added: AVS, CPOTHER, MACRO, and ROUTINE.

LISTFILE

New options: ALLFILE, AUTHFILE; SHARE, SEARCH; HEADER, NOHEADER.

NUCXMAP

New operand added - *name*. New options added are: NOSEGMENT, SEGMENT, ATTRIBUTES, SEGINFO. The new options allow you to specify a nucleus extension that resides in a loaded saved segment.

QUERY

ACCESSED, ALIAS, AUTHORITY, COMDIR, CSLLIB, DISK, ENROLL, FILEPOOL, FILEWAIT, KEYPROTECT, LIMITS, LOADAREA, LOCK, NAMEDEF, SEGMENT, SERVER, and STORECLR.

RDR

New options: MSGSUBS, MSGALL

READCARD

The default options have been changed. The new default options are MINPROMPT and NOREPLACE.

RELEASE

Addition of *dirid* to release an accessed SFS directory.

SET

COMDIR, FILEPOOL, FILEWAIT, FULLSCREEN, KEYPROTECT, LOADAREA, SERVER, STORECLR, and THRESHOLD

XEDIT

New options: LOCK, NOLOCK.

Integration of Between-Release Support Information of VM/SP Release 6

1. VM/SP Enhancements to the IBM Enhanced Connectivity Facilities for VM/System Product, GC24-5295.
2. VM/SP 9370 Processors, 9332 and 9335 Direct Access Storage Devices, and 9347 Tape Drive, GC24-5315.
3. VM Productivity Aids National Language Support Enhancement, GC24-5400.

Summary of Changes

for SC19-6209-4

for VM/SP Release 5

New Commands for Release 5 of VM/SP

The following CMS commands are new for this release.

ALARM VSCREEN

Sound the terminal alarm when the display is refreshed.

CLEAR VSCREEN

Erases data in the virtual screen by overwriting the data buffer with nulls.

CLEAR WINDOW

Scrolls past all data in the virtual screen to which the window is connected so that no scrollable data is displayed in the window.

CMSSERV

Starts IBM Cooperative Processing communications on VM/SP.

CONVERT COMMANDS

Converts a CMS file containing Command Syntax Definition Language (CSDL) statements into an internal form for the parsing facility.

CURSOR VSCREEN

Positions the cursor on specified line and column in a virtual screen.

DEFINE VSCREEN

Create a virtual screen.

DEFINE WINDOW

Creates a window with the specified name, size, and position on the physical screen.

DELETE VSCREEN

Removes a virtual screen definition.

DELETE WINDOW

Removes a window definition.

DROP WINDOW

Moves a window down in the order of displayed windows.

GENMSG

Converts a message repository file into an internal form.

GET VSCREEN

Writes data from a CMS file to the specified virtual screen.

HIDE WINDOW

Prevents the specified window from being displayed, and, optionally, connects the window to a virtual screen.

MAXIMIZE WINDOW

Expands a window to the physical screen size.

MINIMIZE WINDOW

Reduces the size of the window to one line.

MOREHELP

Obtains either additional or related information about the latest valid HELP command you issued.

PARSECMD

Calls the parsing facility from within an exec.

POP WINDOW

Moves a window up in the order of displayed windows.

POSITION WINDOW

Changes the location of a window on the physical screen.

PUT SCREEN

Copies the physical screen and writes the image to a CMS file.

PUT VSCREEN

Writes the data from the scrollable data area of a virtual screen to a CMS file.

RESTORE WINDOW

Return a maximized or minimized window to its size and location prior to the maximize or minimize.

ROUTE

Directs data of a particular message class to a virtual screen.

SCROLL

Moves a window to a new location on the virtual screen to which it is connected.

SHOW WINDOW

Places a window on top of all other displayed windows and connects the window to a virtual screen.

SIZE WINDOW

Changes the number of lines and columns for a specified window.

VALIDATE

Verifies the syntax of a file identifier and verifies whether or not a disk is accessed.

WAITREAD VSCREEN

Used from an EXEC, updates the virtual screen with data, refreshes the physical screen, and waits for the next attention interrupt.

WAIT VSCREEN

Updates the virtual screen with data.

WRITE VSCREEN

Enters information in a virtual screen. Information is queued to a virtual screen and is displayed the next time the screen is refreshed.

XMITMSG

Retrieves a message from a CMS message repository file or your own message repository file.

Border Commands

Manipulate windows when entered on the corners of window borders. Several single-character border commands have been added to manipulate windows in the full-screen CMS environment:

B	Scrolls the window backward
C	Clears the window of scrollable data
D	Drops the window
F	Scrolls the window forward
H	Hides the window
L	Scrolls the window to the left
M	Changes the location of the window
N	Minimizes the window
O	Restores the window
P	Pops the window
R	Scrolls the window to the right
S	Changes the size of the window
X	Maximizes the window

Modified Commands for Release 5 of VM/SP

The following CMS commands have been modified this release.

ACCESS

Addition of the SAVEONLY and NOSAVE options that allow you to specify whether a disk is accessed with a saved copy of the file directory or with a file directory in user storage.

DEFAULTS

Defaults options for HELP can be set for a BRIEF or DETAIL layer of information.

EXECDROP

Addition of the SHARED option that drops an exec that is located in an Installation DCSS.

EXECIO

Addition of the BUFFER option allows you to specify the length of the CP command response expected from a CP operation. The length may be from 1 to 2³¹-1 characters (bytes).

EXECMAP

Addition of the SHARED option allows you to list execs that are located in an Installation DCSS.

EXECUPDT

Comments can be removed from an exec source file with the COMMENTS option. The ETMODE option specifies that the source file contains DBCS characters and that shift-in and shift-out should be paired while comments are removed.

FILEDEF

Addition of the ALT option allows you to specify an alternate tape drive. The SYSPARM option is added for OS simulation standard label tape processing exits.

FORMAT

The BLKSIZE option defaults to a block size that optimizes the I/O and data storage for the particular device.

GLOBAL

The GLOBAL command has been enhanced to allow you to list up to 63 libraries from one of the supported library types.

HELP

Addition of the BRIEF, DETAIL, and RELATED options allow to specify different levels, or layers, of information. The EXTEND option uses the HELP search order when you issue the HELP command from the editing environment and the component is not specified.

INCLUDE

The HIST option saves the history information (comments) from text files. The NOHIST does not save the history information from text files.

LOAD

The HIST option saves the history information (comments) from text files. The NOHIST does not save the history information from text files.

PRINT

Addition of the OVERSIZE option allows you to print files that have records larger than the carriage size of the virtual printer and files that have a SPECIAL status of YES.

QUERY

APL, BORDER, CHARMODE, CMSPF, CURSOR, DISPLAY, DOSLIB, FULLREAD, FULLSCREEN, HIDE, INSTSEG, KEY, LIBRARY, LOADLIB, LOCATION, LOGFILE, MACLIB, NONDISP, REMOTE, RESERVED, SHOW, TEXT, TXTLIB, TRANSLATE, VSCREEN, WINDOW, and WMPF.

SET

ABBREV, APL, BORDER, CHARMODE, CMSPF, FULLREAD, FULLSCREEN, INSTSEG, LANGUAGE, LANGLIST, LINEND, LOCATION, LOGFILE, NONDISP, REMOTE, RESERVED, TEXT, TRANSLATE, VSCREEN, WINDOW, and WMPF.

TXTLIB

The FILENAME option indicates that all the filenames specified will be used as the member names for their respective entries in the TXTLIB file instead of the first CSECT in the file's text deck.

UPDATE

The UPDATE command is enhanced to list up to 29 macro library (MACLIB) filenames.

XEDIT

The WINDOW option specifies the window and virtual screen that the System Product editor uses to display the file being edited.

Migration of CMS Commands into the Nucleus

The following CMS commands are now nucleus resident commands:

- COPYFILE
- GLOBALV
- IDENTIFY
- PRINT

CMS Command Search Order

Document the search for translations and abbreviations in CMS Command Search Order description.

CMS Command Syntax Error Messages

For some CMS commands, syntax error messages are issued by the CMS Parsing Facility.

Integration of Functional Enhancements to Release 4

Document support of the following:

- IBM 3380 Direct Access Storage Device, Models AE4 and BE4
- IBM 3380 Direct Access Storage Device Support for VSE/VSAM
- CMS Vector Processing
- CMS Loader

Miscellaneous

- This major revision incorporates minor technical and editorial changes.
- A *keyword* operand is added to the .CS HELPCONV format word
- The DDR command is moved to the *VM/SP Administration*.

Summary of Changes for SC19-6209-3 for VM/SP Release 4

New Commands for Release 4 of VM/SP

The following CMS commands are new for this release.

EXECDROP

Purges storage resident EXECs.

EXECLOAD

Load EXECs into storage.

EXECMAP

Provides a list of storage resident EXECs.

EXECSTAT

Provides the status of a specific EXEC.

HELPCONV

Converts a specified file into a formatted HELP file, leaving the .CS, .CM, and .MT control words in the file.

MACLIST

Displays a list of all members in a specified MACLIB, with the ability to edit and issue commands from the list.

Commands and Macros that have been Modified

The following commands and macros have been modified for this release.

DDR command

Addition of the MODE 38K option for use with the 3480 Magnetic Tape Subsystem. Supports the use of the COMPACT option on the OUTPUT control statement for the DUMP function.

DEFAULTS command

Defaults options can be set up for the HELP and MACLIST commands.

EXECIO

Addition of the VAR and STEM options that allow you to use the EXECIO command directly with REXX or EXEC 2 variables.

FILEDEF

Addition of the 18TRACK option for the TAPn operand.

HELP command

Addition of MESSAGE, MSG, and TASKS operands. Addition of DESCRIPTION, FORMAT, PARMS, OPTIONS, NOTES, ERRORS, SCREEN, and NOSCREEN options that allow you to select specific areas of a command HELP file to be displayed.

INCLUDE command

Addition of the RLDSAVE option that instructs the CMS loader to save the relocation information from text files.

LABELDEF command

Addition of VOLID ? and VOLID SCRATCH operands.

LOAD

Addition of the RLDSAVE option that instructs the CMS loader to save the relocation information from text files.

MACLIB command

Individual member names can now be specified with the MAP operand. Addition of the STACK, LIFO, FIFO, and XEDIT options that allow you to stack the MAP output.

TAPE

Addition of the 18TRACK option that allows you to specify an 18 track tape that is used with the 3480 Magnetic Tape Subsystem. The TRANSFER IMMEDIATE and TRANSFER BUFFERED options specify the write mode for the 3480 Magnetic Tape Subsystem.

XEDIT command

New MEMBER option allows you to specify the name of a macro library member to be edited.

TAPECTL macro

Addition of the BLKBUFF operand to be used with the LOCBLK and RDBLKID functions for the 3480 Magnetic Tape Subsystem.

WRTAPE

ADDITION of TRAN BUFF and TRAN IMMED operands to specify the write mode when used with the 3480 Magnetic Tape Subsystem.

Migration of CMS Commands into the Nucleus

The following CMS commands are now nucleus resident commands:

- ACCESS
- DLBL
- FILEDEF
- RELEASE
- SET

CMS Command Search Order

Document the search for storage resident EXECs in the CMS Command Search Order description.

Integration of Functional Enhancements to Release 3

Document support of the following:

- IBM 3800 Printing Subsystems, Models 1 and 3.
- IBM 3370 Direct Access Storage Device, Models A2 and B2.
- 3290 Information Panel.
- IBM 4248 Printer.

New VM/SP Component

Document support of the Interactive Problem Control System (IPCS) component of VM/SP.

Miscellaneous

- This major revision incorporates minor technical and editorial changes.
- Document support of the IBM 3480 Magnetic Tape Subsystem
- Document changes to XEDIT screens for messages that appear in mixed case.
- Document the new .MT (MENU TYPE) HELPCONV format word.

Other Changes

- MOVE EDIT SUBCOMMANDS to appendix
- MOVE EXEC CONTROL STATEMENTS to appendix
- Book now has 6 chapters (formerly referred to as sections).
 1. INTRO
 2. CMS Commands
 3. CMS Functions
 4. CMS Macros
 5. Help Format Words
 6. DEBUG subcommands
- and six appendixes
(EDIT SUBCOMMANDS and EXEC control statement have been moved here).

Glossary of Terms and Abbreviations

Some of the following convenience terms are used throughout this publication:

- Throughout this publication, the term “VM/SP” refers to the VM/SP program package when you use it in conjunction with VM/370 Release 6. The terms “CP,” “CMS,” and “IPCS” refer to the VM/370 components enhanced by the functions included in the VM/SP package. Any references to “RSCS,” unless otherwise noted, are to the VM/370 component unchanged by the VM/SP package.

When you install and use VM/SP in conjunction with the VM/370 Release 6 System Control Program (SCP), it becomes a functional operating system that provides extended features to the following components of VM/370 Release 6:

- Control Program (CP)
- Conversational Monitor System (CMS)
- Interactive Problem Control System (IPCS).

VM/SP adds no additional functions to the Remote Spooling Communications Subsystem (RSCS) component of VM/370. However, you can appreciably expand the capabilities of these components in a VM/SP system by installing the RSCS Networking program product (5748-XP1). Note that VM/SP has an enhanced interactive problem control system (VM/SP IPCS) component. This component replaces the unmodified VM/370 interactive problem control system. Detail of this major component are found in the *VM/SP Interactive Problem Control System Guide*, SC24-5260.

Note: For VM/SP users, VM/SP IPCS is more effective than the IPCS Extension Program Product (5748-SA1).

- The term “CMS/DOS” refers to the functions of CMS that become available when you issue the command:
set dos on

CMS/DOS is a part of the normal CMS system, and is not a separate system. Users who do not use CMS/DOS are sometimes referred to as OS users, since they use the OS simulation functions of CMS.
- Unless otherwise noted, the term “VSE” refers to the combination of the DOS/VSE system control program and the VSE/Advanced Functions program product.

In certain cases, the term DOS is still used as a generic term. For example, disk packs initialized for use with VSE or any predecessor DOS or DOS/VS system may be referred to as DOS disks.

The DOS-like simulation environment provided under the CMS component of the VM/System Product, continues to be referred to as CMS/DOS.

- The term “GAM/SP,” Graphic Access Method/System Product, refers to IBM Program Product 5668-978.
- The term “CMS files” refers exclusively to files that are in the format used by CMS file system commands. VSAM and OS data sets and DOS files are not compatible with the CMS file format and cannot be manipulated using CMS file system commands.
- The terms “disk” and “virtual disk” are used interchangeably to indicate disks that are in your CMS virtual machine configuration. Where necessary, a distinction is made between the CMS-formatted disks and disks in OS or DOS format.
- The term “CMS console stack” refers to the combination of the program stack and the terminal input buffer.

A

abend. (1) Abnormal end of task. (2) Synonym for *abnormal termination*.

abend dump. The contents of main storage, or part of main storage, written to an external medium for debugging an error condition that resulted in the termination of a task before its regular completion.

abnormal end of task (abend). Termination of a task before its completion because of an error condition that cannot be resolved by recovery facilities while the task is executing.

abnormal termination. The ending of processing before planned termination. Synonymous with *abend*.

accept. Allowing a connection to the user’s virtual machine from another virtual machine or from the user’s own virtual machine.

access mode. A method VM/SP uses to control user access to data files. Access modes let the user read and write data to a file, or only read data from a file. See *file mode*.

active work unit. A work unit that has uncommitted work associated with it. A request was made on the work unit (other than an atomic request) and no commit or rollback has occurred.

ADCON. An A-type address constant used in calculating storage addresses.

Advanced Program-to-Program Communications (APPC). The inter-program communication service within SNA LU 6.2 on which the APPC/VM interface is based.

Advanced Program-to-Program Communications/VM (APPC/VM). An API for communicating between two virtual machines that is mappable to the SNA LU 6.2 APPC interface and based on IUCV functions. Along with the TSAF virtual machine, APPC/VM provides this communication within a single system and throughout a collection of systems.

alias. A pointer to a base file. An alias can be in the same directory as the base file or in a different directory. There must always be a base file for the alias to point to. The alias references the same data as the base file. Data is not moved or duplicated.

alphameric. Synonym for *alphanumeric*.

alphanumeric. Pertaining to a character set that contains letters, digits, and usually other characters, such as punctuation marks; synonymous with *alphameric*.

APAR. Authorized program analysis report.

APPC. Advanced Program-to-Program Communications.

APPC/VM. Advanced Program-to-Program Communications/VM.

APPC/VM VTAM Support (AVS). A component of VM/SP that lets application programs using APPC/VM communicate with programs anywhere in a network defined by IBM's SNA. AVS transforms APPC/VM into APPC/VTAM protocol.

application program. A program written for or by a user that applies to the user's work, such as a program that does inventory control or payroll.

apply. When servicing a product or component, to generate an auxiliary control structure from a PTF.

area. A term acceptable for DASD space when there is no need to differentiate between space on count-key-data devices and FB-512 devices. See *DASD space*.

assembler language. A source language that includes symbolic machine language statements in which there is a one-to-one correspondence with instruction formats and data formats of the computer.

atomic request. A CMS command or program function that cannot be entered or run on an active work unit.

The work unit must be inactive. When it completes, an atomic request leaves the work unit inactive.

attention interrupt. An I/O interrupt caused by a terminal user pressing the attention key (or equivalent). See *attention key (ATTN key)* and *signaling attention*.

attention key (ATTN key). A function key on terminals that, when pressed, causes an I/O interruption in the processing unit. See *signaling attention*.

ATTN key. Attention key.

authority. In SFS, the permission to access a file or directory. You can have read authority or write authority (which includes read authority). You can also have file pool administration authority, which is the highest level of authority in a file pool.

authorized program analysis report (APAR). An official request to the responsible IBM Change Team to look into a suspected problem with IBM code or documentation. APARs describe problems giving conditions of failure, error messages, abend codes, or other identifiers. They also contain a problem summary and resolution when applicable. See *program temporary fix (PTF)*.

AUX file. Auxiliary control file.

auxiliary control file (AUX file). A file that contains a list of file types of update files applied to a particular source file or to control the service level used during build. See *control file* and *preferred auxiliary file*. Synonymous with *auxiliary file*.

auxiliary directory. In CMS, an extension of the CMS file directory for a minidisk, which contains the names and locations of certain CMS modules not included in the minidisk's CMS file directory.

auxiliary file. Synonym for *auxiliary control file*.

AVS. APPC/VM VTAM Support.

B

base file. The first occurrence of a file. It remains the base for the life of the file, even if the file has been renamed. Aliases point to base files.

basic control (BC) mode. A mode in which additional System/370 features, such as new machine instructions, are not operational. Contrast with *extended control (EC) mode*.

basic sequential access method (BSAM). An access method for storing or getting data blocks in a continuous sequence (using either a sequential access or direct access device).

binary digit. Either of the digits 0 or 1 when used in the pure binary numeration system. Synonymous with *bit*.

bit. (1) Either of the binary digits 0 or 1. See *byte*.
(2) Synonym for *binary digit*.

block. A unit of DASD space on FB-512 devices. For example, FB-512 devices can be the IBM 9335, 9332, 9313, 3370, and 3310 DASD using fixed-block architecture.

border. A boundary around a window. The user can enter one-letter BORDER commands from the corners of the border. For example, the letter *P* entered from a border corner pops the window. The border corners are indicated by a + (plus) sign.

bpi. Bits per inch.

Bpi. Bytes per inch.

BSAM. Basic sequential access method.

buffer. An area of storage, temporarily reserved for performing input or output, into which data is read, or from which data is written.

build. In reference to installation and service of a product, to do the necessary steps to produce executable code or systems. This is often called the *build process*.

built-in function. A specialized function, invoked by a keyword, that has been built into the system program code because it is commonly required by many users.

byte. A unit of storage, consisting of eight adjacent binary digits that are operated on as a unit and constitute the smallest addressable unit in the system.

C

callable services library (CSL). A package of CMS assembler routines that can be stored as an entity and made available to application programs.

catalog storage group. The storage group in a file pool that contains information about the objects (such as files and directories) and authorizations that exist in the file pool. See *file pool catalog*.

CC. Condition code.

CCW. Channel command word.

changes. In reference to installation and service, IBM and original equipment manufacturer (OEM) supplied service for their programs. In the IBM service process, there are many ways users can receive information they need to fix (change) a portion(s) of a product they are

running on a VM system. These include PTFs, APARs, user modifications, and information received over the phone. All these types of information are called *changes*.

channel. A path in a system that connects a processor and main storage with an I/O device.

channel command word (CCW). A doubleword at the location in main storage specified by the channel address word. One or more CCWs make up the channel program that directs data channel operations.

channel status word (CSW). An area in storage that provides information about the termination of I/O.

character delete symbol. Synonym for *logical character delete symbol*.

checkpoint. An internal file pool server operation during which the changes recorded on the log minidisks are permanently made to the file pool.

circumventive service. Information that IBM supplies over the phone or on a tape to circumvent a problem by disabling a failing function until a PTF is available to be shipped as a corrective service fix. See *patch* and *zap*.

CKD. Count-key-data.

class authority. Privilege assigned to a virtual machine user in the user's directory entry; each class specified allows access to a subset of all the CP commands. See *privilege class* and *user class restructure (UCR)*.

class B user. See *system resource operator privilege class*.

CMS. Conversational Monitor System.

CMS batch facility. A facility that lets the user run time-consuming or noninteractive CMS jobs in another CMS virtual machine dedicated to that purpose, thus freeing the user's own terminal and virtual machine for other work.

CMSDOS. The standard name of the CMS/DOS saved segment. See *saved segment*.

CMS/DOS. The functions of CMS that become available when the user enters the command: SET DOS ON. CMS/DOS is a part of the regular CMS system and is not a separate system. Users who do not use CMS/DOS are sometimes called OS users, because they use the OS simulation functions of CMS. Synonymous with *DOS simulation under CMS*. Contrast with *OS simulation under CMS*.

CMS/DOS phase library. Synonym for *DOSLIB library*.

CMS editor. A CMS facility that lets the user create, change, insert, delete, or rearrange lines of data in a CMS file. See *edit mode* and *input mode*.

CMS EXEC. An EXEC procedure or EDIT macro written in the CMS EXEC language and processed by the CMS EXEC processor. Synonymous with *CMS program*.

CMS EXEC language. A general-purpose, high-level programming language, particularly suitable for EXEC procedures and EDIT macros. The CMS EXEC processor executes procedures and macros (programs) written in this language. Contrast with *EXEC 2 language* and *Restructured Extended Executor (REXX) language*.

CMS EXEC processor. The component of the VM/SP operating system that interprets and executes procedures and EDIT macros written in the CMS EXEC language.

CMS file directory. A directory on each CMS disk that contains the name, format, size, and location of each of the CMS files on that disk. When a disk is accessed by the ACCESS command, its directory is read into virtual storage and identified with any letter from A through Z. Synonymous with *master file directory block* and *minidisk directory*.

CMS files. Refers exclusively to files in the fixed-block format used by CMS file system commands. VSAM and OS data sets and DOS files are not compatible with the CMS file format and cannot be manipulated using CMS file system commands.

CMS file system. A way to create files in the CMS system. CMS files are created by using an identifier consisting of three fields: file name, file type, and file mode. These files are unique to the CMS system and cannot be read or written using other operating systems.

CMS nucleus. The portion of CMS that is resident in the user's virtual storage whenever CMS is executing. Each CMS user receives a copy of the CMS nucleus when the user IPLs CMS. See *saved system* and *shared segment*.

CMS program. Synonym for *CMS EXEC*.

CMSSERV. A command that starts a CMS router in the Enhanced Connectivity Facilities environment of VM/SP.

CNTRL file. Control file with file type CNTRL.

collection. See *TSAF collection*.

command. A request from a user at a terminal for the execution of a particular CP, CMS, IPCS, GCS, TSAF, or AVS function. A CMS command can also be the name of a CMS file with a file type of EXEC or

MODULE. See *subcommand* and *user-written CMS command*.

command abbreviation. A short form of the command name, operand, or option that is not a truncation of the word. For example, MSG instead of MESSAGE, RDR instead of READER. Contrast with *truncation*.

command line. The line at the bottom of display panels that lets a user enter commands or panel selections. It is prefixed by an arrow (= = = >).

command privilege class. See *privilege class*.

commit. Permanently changing a resource (such as a file or a data base object).

communication link. Synonym for *data link*.

communications directory. A CMS facility that lets APPC/VM applications connect to a resource using symbolic destination names and special NAMES files.

compile. To translate a program written in a high-level programming language into a machine language program.

component. A collection of elements that together form a separate functional unit. A product may contain many components (for example, VM/SP has components of CP, CMS, GCS, TSAF, IPCS, AVS, and Procedures Language/ VM). A component can be part of many products. (CP spans both VM/SP and VM/HPO products.)

component override. Synonym for *component parameter override*.

component parameter override. A component parameter, defined in a component override area, that updates or replaces a component parameter defined in a component area of the product parameter file. Synonymous with *component override* and *override*.

condition code (CC). A code that reflects the result of a previous I/O, arithmetic, or logical operation.

connect. Establishing a path to communicate with another virtual machine or with the user's own virtual machine.

console. A device used for communications between the operator or maintenance engineer and the computer.

console spooling. Synonym for *virtual console spooling*.

console stack. Refers collectively to the program stack and the terminal input buffer.

control block. A storage area that a computer program uses to hold control information.

control data. In reference to a file pool, the data that controls the DASD space and objects within a file pool. Control data consists of the POOLDEF file, the control minidisk, and all minidisks allocated to storage group 1.

control file. (1) In service, a file with file type CNTRL that contains records that identify the updates to be applied and the macro libraries, if any, needed to assemble that source program. (2) A CMS file that is interpreted and directs the flow of a certain process through specific steps. For example, the control file could contain installation steps, default addresses, and PTF prerequisite lists as well as many other necessary items.

control minidisk. In a file pool, the minidisk that tracks the physical DASD blocks allocated to the file pool.

control program. A computer program that schedules and supervises the program execution in a computer system. See *Control Program (CP)*.

Control Program (CP). A component of VM/SP that manages the resources of a single computer so multiple computing systems appear to exist. Each virtual machine is the functional equivalent of an IBM System/370.

control section (CSECT). The part of a program specified by the programmer to be a relocatable unit, all elements of which are loaded into adjoining main storage.

control statement. A statement that controls or affects program execution in a data processing system.

control unit. A device that controls I/O operations at one or more devices.

control unit terminal (CUT). An operational mode that allows one logical terminal session. Contrast with *distributed function terminal (DFT)*.

conversation. A connection between two transaction programs over an LU-LU session that lets them communicate with each other while processing some transaction. The programs establish a conversation, send and receive data in the conversation, and then terminate the conversation.

Conversational Monitor System (CMS). A virtual machine operating system and component of VM/SP that provides general interactive time sharing, problem solving, program development capabilities, and operates only under the control of the VM Control Program (CP).

copy file. A file having file type COPY that contains nonexecutable real storage definitions that are referred to by macros and assemble files.

copy function. The function initiated by a PF key to copy the contents of a display screen onto an associated hardcopy printer. A remote display terminal copies the entire contents of the screen onto a printer attached to the same control unit. A local display terminal copies all information from the screen, except the screen status information, onto any printer attached to any local display control unit.

corrective service. Service that IBM supplies on tape to correct a specific problem.

count-key-data (CKD) device. A disk storage device that stores data in the format: count field, usually followed by a key field, followed by the actual data of a record. The count field contains the cylinder number, head number, record number, and the length of the data. The key field contains the record's key (search argument).

CP. Control Program.

CP command. A command available to all VM users. Class G CP commands let the general user reconfigure their virtual machine, control devices attached to their virtual machine, do input and output spooling functions, and simulate many other functions of a real computer console. Other CP commands let system operators, system programmers, system analysts, and service representatives manage the resources of the system.

CP directory. Synonym for *VM/SP directory*.

CP-owned disk. Any disk formatted by the CP Format/Allocate program and designated as system-owned during system generation; for example, the CP system residence volume, or any disk that contains CP paging, spooling, saved systems, or temporary disk space.

CPTRAP. A CP debugging tool that creates a reader spool file of selected trace table entries, CP data, and virtual machine data in the order that they happen. The IPCS commands can help the user access and print this collected data.

CSECT. Control section.

CSL. Callable services library.

CSL routine. An assembler program that resides in a CSL and that can be invoked from an application program to do a specific function.

CSW. Channel status word.

CUT. Control unit terminal.

cylinder. In a disk pack, the set of all tracks with the same nominal distance from the axis about which the disk pack rotates.

D

DASD. Direct access storage device.

DASD Dump Restore (DDR) program. A service program that copies all or part of a minidisk onto tape, loads the contents of a tape onto a minidisk, or sends data from a DASD or from tape to the virtual printer.

DASD space. (1) Area allocated to DASD units on CKD devices. (2) Area allocated to DASD units on FB-512 devices. Note that *DASD space* is synonymous with *cylinder* when there is no need to differentiate between CKD devices and FB-512 devices. This term applies to VM/370, VM/SP and VM/SP HPO program products.

data control block (DCB). A control block access method routines use to store and retrieve data.

data link. The equipment and rules (protocols) used for sending and receiving data. Synonymous with *communication link*.

data stream. A set of logical records sent one after the other.

DBCS. Double-byte character set.

DCB. Data control block.

dedicated maintenance mode. In reference to a file pool server machine, a mode of file pool server processing during which the file pool server machine has exclusive use of the file pool. The file pool is unavailable to other users. Contrast with *multiple user mode*.

default operand. An operand that has a preset value if a value is not specified on the CP or CMS command line.

delimiter. (1) A flag that separates and organizes items of data. Synonymous with *separator*. (2) A character that groups or separates words or values in a line of input. Usually one or more blank characters separate the command name and each operand or option in the command line. In certain cases, a tab, left parenthesis, or backspace character can also act as a delimiter.

DFT. Distributed function terminal.

direct access storage device (DASD). A storage device in which the access time is effectively independent of the location of the data.

directory. See *auxiliary directory*, *CMS file directory*, *SFS directory*, or *VM/SP directory*.

directory identifier (dirid). A fully-qualified directory name (in which the file pool ID and user ID can be

allowed to default), a file mode letter, or plus (+) or minus (-) file mode syntax (used in commands).

directory name (dirname). A fully-qualified directory name that can incorporate a period (.) to indicate the user's own top directory (used in commands).

dirid. Directory identifier.

dirname. Directory name.

disconnect mode. The mode of operation in which a virtual machine is executing without a physical line or terminal connected as an operator console. Any attempt to issue a read to the console causes the virtual machine to be logged off after 15 minutes have elapsed, unless the user logs on again within the 15-minute interval. Note that with the SCIF, a user can be disconnected from a primary virtual console but still have console communications through the console of the secondary user.

discontiguous saved segment. One or more 64K segments of storage that were previously loaded, saved, and assigned a unique name. The segment(s) can be shared among virtual machines if the segment(s) contains reentrant code. Discontiguous segments used with CMS must be loaded into storage at locations above the address space of a user's CMS virtual machine. They can be detached when no longer needed.

disk. A magnetic disk unit in the user's CMS virtual machine configuration. Also called a virtual disk.

disk operating system (DOS). An operating system for computer systems that use disks and diskettes for auxiliary storage of programs and data.

Disk Operating System/Virtual Storage Extended (DOS/VSE). An operating system that is an extension of DOS/VS. A VSE system consists of: (a) licensed VSE/Advanced Functions support, and (b) any IBM-supplied and user-written programs required to meet the data processing needs of a user. VSE and the hardware it controls form a complete computing system.

display device. An I/O device that gives a visual representation of data.

display mode. A type of editing at a display terminal in which an entire screen of data is displayed at once and in which the user can access data through commands or by using a cursor. Contrast with *line mode*.

display terminal. A terminal with a component that can display information on a viewing surface such as a CRT or gas panel.

distributed function terminal (DFT). An operational mode that allows multiple concurrent logical terminal sessions. Contrast with *control unit terminal (CUT)*.

DOS. Disk operating system.

DOSLIB library. A CMS library that contains the executable phases produced by the DOS Linkage Editor under CMS. These phases are equivalent to, but not usable in the DOS/VS core image library. These phases can be fetched and executed only under CMS/DOS. Synonymous with *CMS/DOS phase library*.

DOS simulation under CMS. Synonym for *CMS/DOS*.

DOS/VSE. Disk Operating System/Virtual Storage Extended.

double-byte character set (DBCS). A character set that requires 2 bytes to uniquely define each character. This contrasts with EBCDIC, in which each printed character is represented by 1 byte.

dump. To write the contents of part or all of main storage, or part or all of a minidisk, to auxiliary storage or a printer. See *abend dump*.

E

EBCDIC. Extended binary-coded decimal interchange code.

EDF. Enhanced disk format.

edit. A function that makes changes, additions, or deletions to a file on a disk. These changes are interactively made. The edit function also generates information in a file that did not previously exist.

EDIT macro. (1) A procedure defined by a frequently used command sequence to do a commonly required editing function. A user creates the macro to save repetitious rekeying of the sequence, and invokes the entire procedure by entering a command (that is, the macro file's file name). The procedure can consist of a long sequence of edit, CMS, and CP commands, along with EXEC 2 or CMS EXEC control statements to control processing within the procedure. (2) A CMS file whose file name starts with a dollar sign (\$) character and whose file type is EXEC.

edit mode. The environment in which CMS EDIT subcommands and System Product Editor (XEDIT) subcommands can be entered by the user to insert, change, delete, or rearrange the contents of a CMS file. Contrast with *input mode*.

enhanced disk format (EDF). A CMS file storage format that supports files consisting of 512-, 1K-, 2K-, or 4K-byte CMS blocks.

entry point. An address or label of an instruction performed upon entering a computer program, a routine, or a subroutine. A program can have several

different entry points, each corresponding to a different function or purpose.

environmental record editing and printing program (EREP). A program that makes the data contained in the system recorder file available for further analysis.

EOF. End of file.

erased alias. An alias that no longer points to a base file because the base file was erased.

EREP. Environmental record editing and printing program.

EXEC 2 EXEC. Synonym for *EXEC 2 program*.

EXEC 2 language. A general-purpose, high-level programming language, particularly suitable for EXEC procedures and XEDIT macros. The EXEC 2 processor runs procedures and XEDIT macros (programs) written in this language. Contrast with *CMS EXEC language* and *Restructured Extended Executor (REXX) language*.

EXEC procedure. (1) A procedure defined by a frequently used sequence of CMS and CP commands to do a commonly required function. A user creates the procedure to save repetitious rekeying of the sequence, and invokes the entire procedure by entering a command (that is, the exec file's file name). The procedure could consist of a long sequence of CMS and CP commands, along with REXX, EXEC 2, or CMS EXEC control statements to control processing within the procedure. (2) A CMS file with a file type of EXEC.

EXEC 2 processor. A program in VM/SP that interprets and executes procedures, EDIT macros, and XEDIT macros written in the EXEC 2 language.

EXEC 2 program. An EXEC procedure, EDIT macro, or XEDIT macro written in the EXEC 2 language and processed by the EXEC 2 processor. Synonymous with *EXEC 2 EXEC*.

explicit lock. A lock on a file or directory that a user explicitly created by entering a CREATE LOCK command or executing a DMSRLOC CSL routine.

extended binary-coded decimal interchange code (EBCDIC). A set of 256 characters, with each character represented by 8 bits.

extended control (EC) mode. A mode in which all features of a System/370 computing system, including dynamic address translation, are operational. Contrast with *basic control (BC) mode*.

extended PLIST (untokenized parameter list). Four addresses that indicate the extended form of a command as it was entered at a terminal.

external security manager. A program that either augments or completely replaces the authorization checking done by file pool server processing.

F

FB-512. An FBA device that stores data in 512-byte blocks (refers to the IBM 9335, 9332, 9313, 3370, and 3310 DASDs).

FBA. Fixed-block architecture.

FCB. (1) Forms control buffer. (2) Function control block.

FIFO (first-in-first-out). A queuing technique in which the next item to be retrieved is the item that has been on the queue for the longest time. Contrast with *LIFO (last-in-first-out)*.

file access mode. A file mode number that designates whether the file can be used as a read-only or read/write file by a user. See *file mode*.

file definition. (1) Equating a CMS file identifier (file name, file type, file mode) with an OS data set name by the FILEDEF command; or equating a DOS file ID with a CMS file identifier by the DLBL command. (2) Identifying the input or output files used during execution of a program (by way of either the FILEDEF or DLBL commands).

file ID. A CMS file identifier that consists of a file name, file type, and file mode. The file ID is associated with a particular file when the file is created, defined, or renamed under CMS. See *file name*, *file type*, and *file mode*.

file mode. A two-character CMS file identifier field comprised of the file mode letter (A through Z) followed by the file mode number (0 through 6). The file mode letter indicates the minidisk or SFS directory on which the file resides. The file mode number indicates the access mode of the file. See *file access mode*.

file name. A one-to-eight character alphanumeric field, comprised of A through Z, 0 through 9, and special characters \$ # @ + - (hyphen) : (colon) _ (underscore), that is part of the CMS file identifier and serves to identify the file for the user.

file pool. A collection of minidisks managed by SFS. It contains user files and directories and associated control information. Many users' files and directories can be contained in a single file pool.

file pool catalog. The part of a file pool that contains information about the objects stored in the file pool and

the authorizations granted on those objects. See *catalog storage group*.

file pool ID. The name of a file pool. It is part of a fully-qualified directory name, identifying where the directory and all files in it are located. It has up to eight characters, followed by a colon (:).

file space. A user's allocation of space within a file pool.

file status table (FST). A table that describes the attributes of a file on a CMS disk, including file name, file type, file mode, date last written, and other status information.

file system control block (FSCB). A collection of information about a CMS file, that CMS OS simulation commands and user programs use. A file system control block is established for a file by the FILEDEF command or the FSCB macro instruction.

file type. A one-to-eight character alphanumeric field, comprised of A through Z, 0 through 9, and special characters \$ # @ + - (hyphen) : (colon) _ (underscore), that is used as a descriptor or as a qualifier of the file name field in the CMS file identifier. See *reserved file types*.

fixed-block architecture (FBA) device. A disk storage device that stores data in blocks of fixed size or records; these blocks are addressed by block number relative to the beginning of the particular file.

forms control buffer (FCB). In the 3800 Printing Subsystem, a buffer for controlling the vertical format of printed output. The FCB is analogous to the punched-paper, carriage-control tape that IBM 1403 Printers use.

free storage. Storage not allocated. The blocks of memory available for temporary use by programs or by the system.

FSCB. File system control block.

FST. File status table.

full-screen CMS. When a user enters the command SET FULLSCREEN ON, CMS is in a window and can take advantage of 3270-type architecture and windowing support, and various classes of output are routed to a set of default windows. Also, users can type commands anywhere on the physical screen and scroll through commands and responses previously displayed. See *windowing*.

function control block (FCB). In Subsystem Support Services (SSS), a control block that contains information such as a function's status, event control block, task I/O queue, and I/O queue.

G

GAM/SP. Graphics Access Method/System Product.

GCS. Group Control System.

general register. In CMS, a register that does operations such as binary addition, subtraction, multiplication, and division. General registers primarily compute and modify addresses in a program.

global resource. A resource accessible from anywhere within a TSAF collection and whose identity is known throughout the collection. A shared file system file pool is an example of a global resource. Contrast with *local resource* and *private resource*.

global resource manager. An application that runs in a server virtual machine and identifies itself to the TSAF collection as a global resource owner using *IDENT. Contrast with *local resource manager* and *private resource manager*.

group. Synonym for *virtual machine group*.

Group Control System (GCS). A component of VM/SP, consisting of a shared segment that the user can IPL and run in a virtual machine. It provides simulated MVS services and unique supervisor services to help support a native SNA network.

H

handshaking feature. See *VM/VS handshaking feature*.

host system. A data processing system that prepares programs and the operating environments for use by another computer or controller.

I

ID card. Under VM/SP, the identification card that indicates the destination user ID of a deck of real cards. These cards are read into the system card reader or into the card reader of an RSCS remote station.

immediate command. A type of CMS command that, when entered after an attention interruption, causes program execution, tracing, or terminal display to stop. Another immediate command can be entered to resume tracing or terminal display. The immediate commands are HB (halt batch execution), HI (halt all System Product Interpreter or EXEC 2 programs or macros), HO (halt tracing), HT (halt typing), HX (halt execution), RO (resume tracing), RT (resume typing), SO (suspend tracing), TE (trace end), and TS (trace start). They are called immediate commands because they are executed as soon as they are entered; they are not stacked in the console stack. Within an exec,

immediate commands can be established or cancelled by the CMS command IMMCMD.

implicit lock. A lock automatically acquired and freed when you run CMS commands and program functions against files or directories that reside in an SFS file pool. Multiple readers and one writer can access the file or directory.

implied CP command. In CMS, a CP command invoked without preceding the command line with CP.

implied EXEC. An EXEC procedure invoked without identifying it as such; that is, the word exec is not used for the invocation. Only the file name is used, as if entering a CMS command.

inactive work unit. A work unit on which no requests have yet been made, or an atomic request was made, or requests were made and have been committed or rolled back; that is, an inactive work unit has no uncommitted work associated with it.

indicator. A 1-byte area of storage that contains either the character "1" to denote a true condition or the character "0" to denote a false condition.

initial program load (IPL). The initialization procedure that causes an operating system to begin operation. A VM user must IPL the specific operating system into the virtual machine that will control the user's work. Each virtual machine can be loaded with a different operating system.

initialize. To set counters, switches, addresses, or contents of storage to starting values.

input line. For typewriter terminals, information keyed in by a user between the time the typing element of the terminal comes to rest following a carriage return until another carriage return is typed. For display terminals, the data keyed into the user input area of the screen. See *user input area*.

input mode. In the CMS Editor or System Product Editor (XEDIT), the environment that lets the user key in new lines of data. Contrast with *edit mode*.

input/output (I/O). (1) Pertaining to a device whose parts can do an input process and an output process at the same time. (2) Pertaining to a functional unit or channel involved in an input process, output process, or both, concurrently or not, and to the data involved in such a process.

input stream. The sequence of job control statements and data submitted (to an operating system) through an input unit especially started for this purpose by the operator.

interactive. The classification given to a virtual machine depending on this virtual machine's processing

characteristics. When a virtual machine uses less than its allocation time slice because of terminal I/O, the virtual machine is classified as being interactive. Contrast with *noninteractive*.

Interactive Problem Control System (IPCS). A component of VM/SP that permits online problem management, interactive problem diagnosis, online debugging for disk related CP or virtual machine abend dumps or CPTRAP files, problem tracking, and problem reporting.

interface. A shared boundary between two or more entities. An interface might be a hardware or software component that links two devices or programs together.

interrupt. A suspension of a process, such as execution of a computer program, caused by an external event and done in such a way that the process can be resumed.

inter-user communication vehicle (IUCV). A VM/SP generalized CP interface that helps the transfer of messages either among virtual machines or between CP and a virtual machine.

invoke. To start a command, procedure, or program.

I/O. Input/output.

IPCS. Interactive Problem Control System.

IPL. Initial program load.

IUCV. Inter-user communication vehicle.

K

K. kilobyte.

kilobyte (K). 1024 bytes.

L

LIFO (last-in-first-out). A queuing technique in which the next item to be retrieved is the item most recently placed in the queue. Contrast with *FIFO* (*first-in-first-out*).

line delete symbol. Synonym for *logical line delete symbol*.

line deletion symbol. Synonym for *logical line delete symbol*.

line end symbol. Synonym for *logical line end symbol*.

line mode. The mode of operation of a display terminal that is equivalent to using a typewriter-like terminal. Contrast with *display mode*.

line number. A number located at either the beginning or the end of a record (line) that can be used during editing to refer to that line. See *prompting*.

link. (1) In RSCS, a connection, or ability to communicate, between two adjacent nodes in a network. (2) In TSAF, the physical connection between two systems.

load. In reference to installation and service, to move files from tape to disk, auxiliary storage to main storage, or minidisks to virtual storage within a virtual machine.

loader. A routine, commonly a computer program, that reads data into main storage.

load map. A map containing the storage addresses of control sections and entry points of a program loaded into storage.

local. Two entities (for example, a user and a server) are said to be local to each other if they belong to the same system within a collection or to the same node within an SNA system. Contrast with *remote*.

local resource. A resource accessible from only within a single VM system and whose identity is known only within a single VM system in the TSAF collection. Contrast with *global resource* and *private resource*.

local resource manager. An application that runs in a virtual machine and identifies itself to the local system in the TSAF collection as a local resource owner by *IDENT. Contrast with *global resource manager* and *private resource manager*.

local service. Changes manually applied to a product or component (that is, not using the program update service or corrective service procedures). See *circumventive service* and *user modification*.

lock. A tool for controlling concurrent usage of SFS objects. Implicit locks are acquired and automatically released when you run CMS commands and program functions in SFS. Explicit locks let you control the type and duration of the lock.

logical character delete symbol. A special editing symbol, usually the *at* (@) sign, that causes CP to delete it and the immediately preceding character from the input line. If many delete symbols are consecutively keyed in, that same number of preceding characters are deleted from the input line. The value can be redefined or unassigned by the installation or the user. Synonymous with *character delete symbol*.

logical line. A command or data line that can be separated from one or more additional command or data lines on the same input line by a logical line end symbol.

logical line delete symbol. A special editing symbol, usually the cent (¢) sign, that causes CP to delete the previous logical line in the input line back to and including the previous logical line end symbol. Synonymous with *line delete symbol* and *line deletion symbol*. See *logical line*.

logical line end symbol. A special editing symbol, usually the pound (#) sign, that lets the user key in the equivalent of several command or data lines in the same physical line; that is, each logical line except the last line is terminated with the logical line end symbol. Synonymous with *line end symbol*.

logical record. A formatted record that consists of a 2-byte logical record length and a data field of variable length.

logical saved segment. A portion of a physical saved segment that CMS can manipulate. Each logical segment can contain different types of program objects, such as modules, text files, execs, callable services libraries, language repositories, user-defined objects, or a single minidisk directory. A system segment identification file (SYSTEM SEGID) associates a logical saved segment to the physical saved segment in which it resides. See *physical saved segment* and *saved segment*.

logical unit (LU). An entity addressable within an SNA-defined network, similar to a node within a VM network. LUs are categorized by the types of communication they support. A TSAF collection in an SNA network is viewed as one or more LUs.

logoff. The procedure by which a user ends a terminal session.

logon. The procedure by which a user begins a terminal session.

look-aside entry. A nucleus resident routine becomes a look-aside entry after it has been executed.

M

machine. A synonym for a virtual machine running under the control of VM/370 or VM/SP.

machine ID. A 2-byte field that uniquely defines a virtual machine within a virtual machine group. Machine ID is sometimes combined with task ID to uniquely identify a task within the virtual machine group.

macro. Synonym for *macrodefinition* and *macroinstruction*.

macrodefinition. A set of statements that defines the name of, format of, and conditions for generating a sequence of assembler language statements from a single source statement. Synonymous with *macro*.

macroinstruction. In assembler language programming, an assembler language statement that causes the assembler to process a predefined set of statements called a macrodefinition. The statements usually produced from the macrodefinition replace the macroinstruction in the program. Synonymous with *macro*.

macro library. A library of macrodefinitions.

map. In CMS, the file that contains a CMS output listing, such as (1) a list of macros in the MACLIB library, including macro size and location within the library, (2) a listing of the directory entries for the DOS/VS system or private source, relocatable, or core image libraries, (3) a linkage editor map for CMS/DOS programs, and (4) a module map containing entry point locations.

master file directory. A directory on each CMS disk that contains the name, format, size, and location of all the CMS files on the disk. When a disk is accessed by the ACCESS command, the directory is read into main storage and identified with one of the 26 disk mode letters (A through Z).

master file directory block. Synonym for *CMS file directory*.

MB. Megabyte.

MDISK. (1) Another name for minidisk. (2) The user directory that describes a user's storage space.

megabyte (MB). 1,048,576 bytes.

merge. When receiving files from a service tape using the VMFREC EXEC, the process of moving existing service files from each minidisk or SFS directory in the target string (as defined by the MERGE tag in the product parameter file) to the minidisk or SFS directory that contains the previous service level. The result is that the primary target minidisk or directory is left empty and ready to receive the latest service from the tape.

message. Data sent from a source application to a target application program in a conversation.

message repository. A source file that contains message texts for a VM component or user application. It is compiled into internal form by the GENMSG command. The message text in a repository file can be translated and used to support national languages.

MIH. Missing interrupt handler.

minidisk. A logical subdivision (or all) of a physical disk pack that has its own virtual device address, consecutive virtual cylinders (starting with virtual cylinder 0), and a VTOC or disk label identifier. Each

user virtual disk is preallocated and defined by a VM/SP directory entry as belonging to a user.

minidisk directory. Synonym for *CMS file directory*.

minimum truncation. The shortest form of a command name, operand, or option that can be keyed in and still be recognized by VM/SP. For example, AC is the minimum truncation for the ACCESS command. However, note that the letter A is the minimum truncation for ASSEMBLE. See *truncation*.

missing interrupt handler (MIH). A VM/SP facility that detects incomplete I/O conditions by monitoring I/O activity. It also tries to correct incomplete I/O conditions without operator intervention.

module. (1) A unit of a software product that is discretely and separately identifiable with respect to modifying, compiling, and merging with other units, or with respect to loading and execution. For example, the input to, or output from, a compiler, the assembler, the linkage editor, or an exec routine. (2) A nonrelocatable file whose external references have been resolved.

multiple user mode. In reference to a file pool server machine, a mode of file pool server processing during which it processes user requests for file pool data. (The CMS FILESERV START command starts multiple user mode processing.) Contrast with *dedicated maintenance mode*.

Multiple Virtual Storage (MVS). An alternative name for OS/VS2.

MVS. Multiple Virtual Storage.

N

namedef. A temporary name that represents either: (1) a file name and file type, or (2) an SFS directory name. Namedefs are used in program functions so that it is not necessary to directly code a file name and file type or directory name in the program.

named system. A system that has an entry in the CP system name table (DMKSNTBL). The entry in the system name table includes the system name and other pertinent data so that the system can later be saved. See *saved system*.

native mode. Refers to running an operating system stand-alone on the real machine instead of under VM/SP.

NCPDUMP. Network control program DUMP.

netdata. The name of the format that sends a file when the NEW option of the CMS SENDFILE command is specified.

network. Any set of two or more computers, workstations, or printers linked in such a way as to let data be transmitted between them.

node. (1) A single processor or a group of processors in a teleprocessing network. (2) A computer, workstation, or printer, when it is participating in a network.

node ID. Node identifier.

node identifier (node ID). The name by which a node is known to all other nodes in a network.

noninteractive. The classification given to a virtual machine depending on this virtual machine's processing characteristics. When a virtual machine usually uses all its allocated time slice, it is classified as being noninteractive or compute bound. Contrast with *interactive*.

nucleus. The part of CP or CMS resident in main storage.

NUCON. The nucleus constant area of CMS.

null line. A logical line with a length of zero that usually signals the CMS Editor to end input mode and enter edit mode. In VM/SP, a null line for typewriter terminals is a terminal input line consisting of a return character as the first and only information, or a logical line end symbol as the last character in the data line. For display devices, a null line is indicated by the cursor positioned at the beginning of the user input area or the data in the user input area ending with a logical line end symbol.

O

object code. Compiler or assembler output that is executable machine code or is suitable for more processing to produce executable machine code. Contrast with *source code*.

object module. A module that is the output of an assembler or a compiler and is input to a linkage editor.

operand. Information entered with a command name to define the data on which a command processor operates and to control the execution of the command processor.

Operating System/Virtual Storage (OS/VS). A family of operating systems that control IBM System/360 and System/370 computing systems. OS/VS includes VS1, VS2, MVS/370, and MVS/XA.

OS simulation under CMS. The environment of CMS that permits the simulation of OS functions. Contrast with *CMS/DOS*.

OS/VS. Operating System/Virtual Storage.

OS/VS2. A virtual storage operating system that is an extension of OS/MVT.

overhead. The additional processor time charged to each virtual machine for the CP functions needed to simulate the virtual machine environment and for paging and scheduling time.

overlay. The technique of repeatedly using the same areas of internal storage during different stages of a program.

override. Synonym for *component parameter override*.

P

pack. A set of flat, circular recording surfaces that a disk storage device uses. A disk pack.

page. A fixed-length block that has a virtual address and can be transferred between real storage and auxiliary storage.

page frame. A block of 4096 bytes of real storage that holds a page of virtual storage.

paging. Transferring pages between real storage and external page storage.

parameter. A variable that is given a constant value for a specified application and that may denote the application.

parameter list (PLIST). In CMS, a string of 8-byte arguments that call a CMS command or function. The first argument must be the name of the command or function to be called. General register 1 points to the beginning of the parameter list.

parent directory. (1) The directory for a CMS disk that has a disk extension defined for it by the ACCESS command. (2) In SFS, the next higher-level directory in which the current directory is defined.

part. A CMS file provided on a product tape or service tape as input to the build process. See *build*. A part is the smallest serviceable unit of a component.

password. In computer security, a string of characters known to the computer system and a user, who must specify it to gain full or limited access to a system and to gain full or limited access to a system and to the data stored within it.

patch. A circumventive service change applied directly to object code in a text deck in a nucleus.

path. In APPC/VM or IUCV, a connection between two application programs that are on the same or different systems. Paths have names assigned to them.

personal computer (PC). A desk-top, floor-standing, or portable microcomputer that usually consists of a system unit, a display monitor, a keyboard, one or more diskette drives, internal fixed-disk storage, and an optional printer.

PF key. Programmed function key.

physical saved segment. One or more pages of storage that have been named and retained on a CP-owned volume (DASD). Once created, it can be loaded within a virtual machine's address space or outside a virtual machine's address space. Multiple users can load the same copy. A physical saved segment can contain one or more logical saved segments. A system segment identification file (SYSTEM SEGID) associates a physical saved segment to its logical saved segments. See *logical saved segment* and *saved segment*.

physical screen. Synonym for *screen*.

physical unit block (PUB). In a VSE system, an entry in a table containing the channel and device address of a device. There is a physical unit block for each physical device available in the system.

PLIST. Parameter list.

preferred auxiliary file. In CMS, an auxiliary file that applies to a particular version of a source module to be updated, if multiple versions of the module exist.

prefix area. The five left-most positions on the System Product Editor's full-screen display, in which prefix subcommands or prefix macros can be entered. See *prefix macros* and *prefix subcommands*.

prefix macros. System Product Editor macros entered in the prefix area of any line on a full-screen display. See *prefix area*.

prefix storage area (PSA). A page zero of real storage that contains machine-used data areas and CP global data.

prefix subcommands. System Product Editor subcommands entered in the prefix area of any line on a full-screen display. See *prefix area*.

preventive service. The massive application of PTFs from the PUT. Contrast with *selective preventive service*.

private resource. A resource accessible from anywhere within a TSAF collection or SNA network and whose identity is known only within a single virtual machine. Contrast with *global resource* and *local resource*.

private resource manager. An application that runs in a server virtual machine and provides a service for connecting programs, but that does not identify itself to the TSAF collection. Contrast with *global resource manager* and *local resource manager*.

privilege class. One or more classes assigned to a virtual machine user in a VM/SP directory entry; each privilege class specified lets a user access a logical subset of the CP commands. There are eight IBM-defined privilege classes that correspond to specific administrative functions. They are:

- Class A - primary system operator
- Class B - system resource operator
- Class C - system programmer
- Class D - spooling operator
- Class E - system analyst
- Class F - service representative
- Class G - general user
- Class H - reserved for IBM use
- Class Any - available to any user.

The privilege classes can be changed to meet the needs of an installation. See *class authority* and *user class restructure (UCR)*.

process. A systematic sequence of operations to produce a specified result. A process is usually logical, not physical.

product. Any separately installable software program, whether supplied by IBM or otherwise, distinct from others and recognizable by a unique identification code. The product identification code is unique to a given product, but does not identify the release level of that product.

PROFILE EXEC. A special EXEC procedure with a file name of PROFILE that a user can create. The procedure is usually executed immediately after CMS is loaded into a virtual machine (also known as IPL CMS).

program function (PF) key. On a terminal, a key that can do various functions selected by the user or determined by an application program.

program stack. Temporary storage for lines (or files) being exchanged by programs that execute under CMS. See *console stack*.

program status word (PSW). An area in storage that indicates the order in which instructions are executed, and to hold and indicate the status of the computer system.

program temporary fix (PTF). Code changes needed to correct a problem reported in an APAR. The corrected code is included in later releases. A PTF contains one or more APAR fixes. For object maintained parts that are changed, the PTF includes replacement parts. For source maintained parts that are changed, the PTF includes update files and replacement parts. Each PTF is unique to a given release of a product. If the same problem occurs in multiple releases of a product, a separate PTF is defined for each release.

program update service. Receiving the contents of a PUT, applying all or some of the changes, and rebuilding the serviced parts. See *preventive service* and *selective preventive service*.

program update tape (PUT). A tape containing a customized collection of service tapes (preventive service) to match the products listed in a customer's ISD (IBM Software Distribution) profile. Each PUT contains cumulative service for the customer's products back to earlier release levels of the product still supported. The tape is distributed to authorized customers of the products at scheduled intervals or on request.

prompt. A displayed message that describes required input or gives operational information.

prompting. An interactive technique that lets the program guide the user in supplying information to a program. The program types or displays a request, question, message, or number, and the user enters the desired response. The process is repeated until all the necessary information is supplied.

PSA. Prefix storage area.

pseudo page fault. A facility available with VM/VS handshaking that lets the VSI virtual machine dispatch another task while waiting for a page-in request to be completed for some other task. Without this facility, the whole virtual machine would wait until the page request was satisfied, even if higher priority tasks were ready to execute.

PSS. Program support services.

PSW. Program status word.

PTF. Program temporary fix.

PUB. Physical unit block.

PUBLIC. In reference to a file pool, all valid users of the system.

PUT. Program update tape.

PVM. VM/Pass-Through Facility.

Q

QSAM. Queued sequential access method.

queued sequential access method (QSAM). An extended version of BSAM. When this method is used, a queue is formed of input data blocks awaiting processing or processed output data blocks awaiting transfer to auxiliary storage or to an output device.

R

read authority. The authority to read the contents of a file without being able to change them. For a directory, read authority lets the user view the names of the objects in the directory.

read-only access. An access mode associated with a virtual disk or SFS directory that lets a user read, but not write or update, any file on the disk or SFS directory.

read/write access. An access mode associated with a virtual disk or SFS directory that lets a user read and write any file on the disk.

real machine. The actual processor, channels, storage, and I/O devices required for VM/SP operation.

receive. (1) Bringing into the specified buffer data sent to the user's virtual machine from another virtual machine or from the user's own virtual machine. (2) To load service files from a service tape.

register. See *general register*.

remote. Two entities (for example, a user and a server) are said to be remote to each other if they belong to different systems within a collection, or to different nodes within an SNA network. Contrast with *local*.

Remote Spooling Communications Subsystem Networking (RSCS). An IBM licensed program and special-purpose subsystem that supports the reception and transmission of messages, files, commands, and jobs over a computer network.

reply. (1) A response to an inquiry. (2) In SNA, a request unit sent only in reaction to a received request unit.

requester. (1) The name given to a virtual machine containing a user program that requests a resource. (2) The program that relays a request to another computer through the SRPI. Contrast with *server*.

reserved file types. (1) File types recognized by the CMS editors (EDIT and XEDIT) as having specific default attributes that include: record size, tab settings, truncation column, and uppercase or lowercase characters associated with that particular file type. The CMS Editor creates a file according to these attributes. (2) File types recognized by CMS commands; that is, commands that only search for and use particular file types or create one or more files with a particular file type.

resource. A program, a data file, a specific set of files, a device, or any other entity or a set of entities that the user can uniquely identify for application program processing in a VM system.

resource manager. An application running in a server virtual machine that directly controls one or more VM resources. There are three categories of VM resource managers: global, local, and private.

response time. (1) The time between the submission of an item of work to a computing system and the return of results. (2) In systems with time sharing, the time between the end of a block or line-end character of terminal input and the display of the first character of system response at the terminal.

Restructured Extended Executor (REXX) language. A general-purpose programming language, particularly suitable for EXEC procedures, XEDIT macros, or programs for personal computing. Procedures, XEDIT macros, and programs written in this language can be interpreted by the System Product Interpreter. Contrast with *CMS EXEC language* and *EXEC 2 language*.

revoked alias. An alias that no longer points to a base file because authorization on the base file was revoked from the owner of the alias. Aliases may also be revoked if the storage group in which the alias resides is restored and the base file, which resides in another storage group, does not exist.

REXX EXEC. An EXEC procedure or XEDIT macro written in the REXX language and processed by the System Product Interpreter. Synonymous with *REXX program*.

REXX language. Restructured Extended Executor language.

REXX program. Synonym for *REXX EXEC*.

rollback. Undoing changes that were made to a resource (such as a file or a data base object).

route. A connection to another system by a logical link and one or more intermediate systems. In TSAF, a number of links and possible intermediate systems that allow the connection of one system to another.

RSCS. Remote Spooling Communications Subsystem Networking.

S

saved segment. A segment of storage that has been saved and assigned a name. The saved segment(s) can be physical saved segment(s) that CP recognizes or logical saved segments that CMS recognizes. The segments can be loaded and shared among virtual machines, which helps use real storage more efficiently, or a private, nonshared copy can be loaded into a virtual machine. See *logical saved segment* and *physical saved segment*.

saved system. A special nonrelocatable copy of a virtual machine's virtual storage and associated registers kept on a CP-owned disk and loaded by name instead of by I/O device address. Loading a saved system by name substantially reduces the time it takes to IPL the system in a virtual machine. In addition, a saved system such as CMS can also share one or more 64K segments of reenterable code in real storage between virtual machines. This reduces the cumulative real main storage requirements and paging demands of such virtual machines.

scale. A line on the System Product Editor's (XEDIT) full-screen display, used for column reference.

SCIF. Single console image facility.

SCP. System control programming.

screen. An illuminated display surface; for example, the display surface of a CRT. Synonymous with *physical screen*.

scrolling. (1) Moving a display image vertically or horizontally in order to view data not otherwise visible within the boundaries of the display screen.
(2) Performing a scroll up, scroll down, scroll right, or scroll left operation.

secondary user. When a user is disconnected — that is, has no virtual console on line — a secondary user can be designated to receive the disconnected user's console messages and to enter commands to the disconnected user's console.

segment. A contiguous 64K or 1024K area of virtual storage (not necessarily contiguous in real storage) allocated to a job or system task. VM/SP does not use 1024K segments, but supports any VM operating system that uses 1024K segments.

selective preventive service. The selective application of PTFs from the PUT. Contrast with *preventive service*.

separator. Synonym for *delimiter*.

server. (1) The general name for a virtual machine that provides a service for a requesting virtual machine.
(2) The program that responds to a request from another computer through SRPI. Contrast with *requester*.

server-requester programming interface (SRPI). (1) A protocol between requesters and servers in an enhanced connectivity network. Includes the protocol to define a cooperative processing subsystem. (2) The interface that enables enhanced connectivity between requesters and servers in a network.

service. Changing a product after installation. See *corrective service*, *local service*, and *program update service*.

session. The SNA term for a connection between two LUs. The LUs involved allocate conversations across sessions.

sever. Ending communication with another virtual machine or with the user's own virtual machine.

SFS. Shared file system.

SFS directory. A group of files. SFS directories can be arranged to form a hierarchy in which one directory can contain one or more subdirectories as well as files.

shared file system (SFS). A part of CMS that lets users organize their files into groups known as *directories* and to selectively share those files and directories with other users.

shared segment. A feature of a saved system or physical saved segment that lets one or more segments of reentrant code or data in real storage be shared among many virtual machines. For example, if a saved CMS system was generated, the CMS nucleus is shared in real storage among all CMS virtual machines loaded by name; that is, every CMS machine's segment of virtual storage maps to the same 64K of real storage. See *discontiguous saved segment* and *saved system*.

SID code. Support Identification code.

signaling attention. An indication that a user has pressed a key or keyed in a CP command to present an attention interrupt to CP or to the user's virtual machine.

simultaneous peripheral operations online (SPOOL). (1) (Noun) An area of auxiliary storage defined to temporarily hold data during its transfer between peripheral equipment and the processor. (2) (Verb) To use auxiliary storage as a buffer storage to reduce processing delays when transferring data between peripheral equipment and the processing storage of a computer.

single console image facility (SCIF). (1) Lets a user, who is disconnected from a primary virtual console, continue to have console communications by way of the console of the secondary user. See *secondary user*.
(2) Enables a virtual machine operator to control multiple virtual machines from one physical terminal.

SIO. Start I/O.

source code. The input to a compiler or assembler, written in a source language. Contrast with *object code*.

source file. A file that contains source statements for such items as high-level language programs and data description specifications.

source update file. A file containing a single change to a statement in a source file. The file can also include requisite information for applying the change. Synonymous with *update file*.

special variable. A reserved variable name assigned a value during processing by the System Product Interpreter, the EXEC 2 processor or CMS EXEC processor. These variables can be tested within an EXEC procedure, edit macro, or XEDIT macro.

SPOOL. Simultaneous peripheral operations online.

spool file class. A one-character class associated with each virtual unit record device. For input spool files, the spool file class lets the user control which input spool files are read next; and, for output spool files, it lets the spooling operator better control or reorder the printing or punching of spool files having similar characteristics or priorities. The spool file class value can be A through Z or 0 through 9.

spool ID. A spool file identification number automatically assigned by CP when the file is closed. The spool ID number can be from 0001 to 9900; it is unique for each spool file. To identify a given spool file, a user must specify the owner's user ID, the virtual device type, and the spool ID.

spooling. The processing of files created by or intended for virtual readers, punches, and printers. The spool files can be sent from one virtual device to another, from one virtual machine to another, and to real devices. See *virtual console spooling*.

SRPI. Server-requester programming interface.

stack. See *console stack* and *program stack*.

stand-alone. Pertaining to an operation independent of another device, program, or system.

storage group. A subset of minidisks within a file pool. Each storage group is identified by a number.

subcommand. The commands of processors such as EDIT or System Product Editor (XEDIT) that run under CMS.

subdirectory. Any directory below a user's top directory. The CREATE DIRECTORY command creates subdirectories. There can be up to eight levels of subdirectories with no limit on the number of them at each level, other than overall DASD space limits. Each level of a subdirectory is an additional identifier of up to 16 characters that is appended to next higher level subdirectory.

supervisor call instruction (SVC). An instruction that interrupts a program being executed and passes control

to the supervisor so that it can do a specific service indicated by the instruction.

SVC. Supervisor call instruction.

synonym. In CMS, an alternative command name defined by the user as equivalent to an existing CMS command name. Synonyms are entries in a CMS file with a file type of SYNONYM. Entering the SYNONYM command allows use of those synonyms until that terminal session ends or until the use of synonyms is revoked by entering the SYNONYM command with no operands.

syntax. The rules for the construction of a command or program.

system administrator. The person responsible for maintaining a computer system.

system control programming (SCP). IBM-supplied programming fundamental to the operation and maintenance of the system. It serves as an interface with IBM licensed programs and user programs and available without additional charge.

System Product Editor. The CMS facility, comprising the XEDIT command and XEDIT subcommands and macros, that lets a user create, change, and manipulate CMS files.

System Product Interpreter. The language processor of the VM/SP operating system that processes procedures, XEDIT macros, and programs written in the REXX language.

system resource operator privilege class. The CP privilege class B user, who controls all the real resources of the machine, such as real storage, disk drives, and tape drives, not controlled by the primary system or spooling operators.

T

target. One of many ways to identify a line to be searched for by the System Product Editor. A target can be specified as an absolute line number, a relative displacement from the current line, a line name, or a string expression.

T-disk. Synonym for *temporary disk*.

temporary disk. An area on a DASD available to the user for newly created or stored files until logoff, at which time the area is released. Temporary disk space is allocated to the user during logon or when entering the CP DEFINE command. Synonymous with *T-disk*.

terminal. A device, usually equipped with a keyboard and a display, capable of sending and receiving information.

terminal input buffer. Holds lines entered at the user's terminal until CMS processes them.

terminal session. The period of time from logon to logoff when a user and the virtual machine can use the facilities of VM/SP or the operating system or both. This also includes any period of time that the virtual machine is running in disconnect mode. See *disconnect mode*.

terminal user. Anyone who uses a terminal to log on to VM/SP.

text deck. An object code file that must be additionally processed to produce executable machine code.

text library. A CMS file that contains relocatable object modules and a directory that indicates the location of each of these modules within the library.

token. An eight-character symbol created by the CMS EXEC processor when it scans an EXEC procedure or EDIT macro statements. Symbols longer than eight characters are truncated to eight characters.

tokenized PLIST (parameter list). A string of doubleword aligned parameters occupying successive doublewords.

top directory. The directory created for a user when the user is enrolled in a file pool. The name of the top directory is the same as the person's user ID.

topmost window. With the window support, the highest window in the display order such that: (1) The window name is not WM or STATUS. (2) The window currently displays at least one virtual screen data line or reserved line. For example, a vsize window connected to a virtual screen such that there are no scrollable data being displayed, is NOT the topmost window.

Note: It may not be obvious by looking at the screen which is the topmost window.

Transparent Services Access Facility (TSAF). A component of VM/SP that handles communication between systems by letting APPC/VM paths span multiple VM systems. TSAF lets a source program connect to a target program by specifying a name that the target has made known, instead of specifying a user ID and node ID.

truncation. A valid shortened form of CP, CMS, GCS, IPCS, RSCS, TSAF (Query only) command names, operands, and options that can be keyed in. When the shortened form is used, the number of key strokes is reduced. For example, the ACCESS command has a minimum allowable truncation of two, so AC, ACC, ACCE, ACCES, and ACCESS are all recognized by CMS as the ACCESS command. Contrast with *command abbreviation*.

TSAF. Transparent Services Access Facility.

TSAF collection. A group of VM processors, each with a TSAF virtual machine, connected by CTC, binary synchronous lines, or LANs.

two-word command. A command resolved to a program name by using the first two tokens of its tokenized parameter list.

typewriter terminal. Printer-keyboard devices that produce hardcopy output only, such as: the IBM 2741 Communication Terminal; the IBM 3215 Console Printer-KeyBoard; the IBM 3767 Communication Terminal, Model 1 or 2, operating as a 2741. This term also refers to the IBM 3101 Display Terminal operating as a 2741.

U

UCR. User class restructure.

universal class card reader. A virtual card reader that can read any class of reader, printer, or punch files spooled or transferred to it.

update file. Synonym for *source update file*.

UPSI. User program switch indicator.

user. Anyone who requests the services of a computing system.

user class restructure (UCR). The extension of the class structure of CP instructions from 8 to 32 classes for each user, command, and diagnose code within the system. This extension allows the installation greater flexibility in authorizing CP instructions.

user ID. User identification.

user input area. On a display device, the lines of the screen where the user is required to key in command or data lines. See *display mode*, *input line*, and *line mode*.

user modification. Any change that a user originates for a product or component.

user program. A transaction program that requests a service from a resource manager program. User programs reside in requester virtual machines.

user program area. In CMS, the virtual storage area occupying location X'20000' to the end of the user's virtual machine. The beginning of the user program area is the default loading point for user programs and for many CMS commands.

user program switch indicator (UPSI). An operand of the CMS SET command. The user can set the switches

(1 byte) to a desired value, which can be tested by a program in CMS/DOS.

user-written CMS command. Any CMS file created by a user that has a file type of MODULE or EXEC. Such a file can be executed as if it were a CMS command by issuing its file name, followed by any operands or options expected by the program or EXEC procedure.

V

vaddr. Virtual address.

variable symbol. In an EXEC procedure, a symbol beginning with an ampersand (&) character, the value of which is assigned by the user, or in some cases by the System Product Interpreter, the EXEC 2 processor, or CMS EXEC processor. The value of a variable symbol can be tested and changed using control statements. See *special variable*.

virtual address. The address of a location in virtual storage. A virtual address must be translated into a real address in order to process the data in processor storage.

virtual card reader. CP's simulation on disk of a real card reader. A virtual card reader can read card, punch, or print records of up to 151 characters in length. The virtual device type and I/O device address are usually defined in the VM/SP directory. See *spool file class* and *universal class card reader*.

virtual console. A console simulated by CP on a terminal such as a 3270. The virtual device type and I/O address are defined in the VM directory entry for that virtual machine.

virtual console spooling. The writing of console I/O on disk as a printer spool file instead of, or in addition to, having it typed or displayed at the virtual machine console. The console data includes messages, responses, commands, and data from or to CP and the virtual machine operating system. The user can invoke or terminate console spooling at anytime. When the console spool file is closed, it becomes a printer spool file. Synonymous with *console spooling*.

virtual disk. A logical subdivision (or all) of a physical disk storage device that has its own address, consecutive storage space for data, and an index or description of the stored data so that the data can be accessed. A virtual disk is also called a minidisk. See *disk*.

virtual machine (VM). A functional equivalent of a real machine.

virtual machine group. The concept in GCS of two or more virtual machines associated with each other through the same named system (for example, IPL GCS1). Virtual machines in a group share common

read/write storage and can communicate with one another through facilities provided by GCS. Synonymous with *group*.

Virtual Machine/System Product (VM/SP). An IBM licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a *real* machine.

virtual printer (or punch). A printer (or card punch) simulated on disk by CP for a virtual machine. The virtual device type and I/O address are usually defined in the VM/SP directory entry for that virtual machine.

virtual = real option. A VM/SP performance option that lets a virtual machine run in VM/SP's virtual = real area. This option eliminates CP paging and, optionally, CCW translation for this virtual machine. Synonymous with $V=R$.

virtual screen. A functional simulation of a physical screen. A virtual screen is a *presentation space* where data is maintained. The user can view pieces of the virtual screen through a window on the physical screen.

virtual storage. Storage space that can be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available, and not by the actual number of main storage locations.

virtual storage access method (VSAM). An access method for direct or sequential processing of fixed and variable-length records on direct access devices. The records in a VSAM data set or file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry-sequence), or by relative-record number.

virtual storage extended (VSE). The generalized term that indicates the combination of the DOS/VSE system control program and the VSE/Advanced Functions program product. Note that in certain cases, the term DOS is still used as a generic term; for example, disk packs initialized for use with VSE or any predecessor DOS or DOS/VS system are sometimes called DOS disks. Also note that the DOS-like simulation environment provided under the VM/SP CMS component and CMS/DOS exists on VM/SP and VM/SP HPO program products and continues to be called CMS/DOS.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in a computer network. It provides single-domain, multiple-domain, and multiple-network capability. VTAM runs under MVS, OS/VS1, VM/SP, and VSE.

VM. Virtual machine.

VMLIB. The name of the CSL supplied with VM/SP and that contains routines to do various VM functions.

VM/Pass-Through Facility. A facility that lets VM users interactively access remote system and processor nodes. These can be remote IBM 4300 processors, other VM systems, with or without this facility installed, or System/370-compatible non-VM systems.

VM/SP. Virtual Machine/System Product.

VM/SP directory. A CP disk file that defines each virtual machine's typical configuration; the user ID, password, regular and maximum allowable virtual storage, CP command privilege class or classes allowed, dispatching priority, logical editing symbols to be used, account number, and CP options desired. Synonymous with *CP directory*.

VM/VS handshaking feature. A communication interface between VM/SP and other operating systems running a virtual machine under VM/SP. These operating systems and CP make each other aware of mutual capabilities and requirements.

valid. Volume identifier.

volume identifier (valid). The volume identification label for a disk.

volume table of contents (VTOC). (1) A table on a direct access volume that describes each data set on the volume. (2) An area on a disk or diskette that describes the location, size, and other characteristics of each file and library on the disk or diskette.

V=R. Synonym for *virtual=real option*.

VSAM. Virtual storage access method.

vscreen. Virtual screen.

VSE. Virtual storage extended.

VTAM. Virtual Telecommunications Access Method.

VTOC. Volume table of contents.

W

window. An area on the physical screen where virtual screen data can be displayed. Windowing lets the user do such functions as defining, positioning, and overlaying windows; scrolling backward and forward through data; and writing data into virtual screens.

windowing. A set of functions that lets the user view and manipulate data in user-defined areas of the physical screen called *windows*. Windowing support lets the user define, position, and overlay windows; scroll backward and forward through data; and write data into virtual screens.

work unit. In CMS, a group of related operations that can be either committed or rolled back as a unit. When the operations associated with a work unit are committed or rolled back, new operations can be associated with the same work unit. These operations can also be committed or rolled back. (The work unit is, in a sense, reusable.) Multiple work units may be active. See *active work unit* and *inactive work unit*.

write authority. The authority to read or change the contents of a file or directory. Write authority implies read authority.

X

XEDIT. See *System Product Editor*.

XEDIT macro. (1) A procedure defined by a frequently used command sequence to do a commonly required editing function. A user creates the macro to save repetitious rekeying of the sequence, and invokes the entire procedure by entering a command (that is, the macro file's file name). The procedure can consist of a long sequence of XEDIT commands and subcommands or both, and CMS and CP commands or both, along with REXX or EXEC 2 control statements to control processing within the procedure. (2) A CMS file with a file type of *XEDIT*.

Z

zap. To modify or dump an individual text file, using the ZAP command or the ZAPTEXT EXEC.

2

2305. Refers to the IBM 2305 Fixed Head Storage Device, Models 1 and 2.

270X. Refers to the IBM 2701, 2702, and 2703 Transmission Control Units or the Integrated Communications Adapter (ICA) on the System/370 Model 135.

2741. Refers to the IBM 2741 Terminal. Information on the 2741 also applies to the IBM 3767 Terminal, unless otherwise noted.

3

3270. Refers to a series of IBM display devices; for example, the IBM 3275, 3276 Controller Display Station, 3277, 3278, and 3279 Display Stations, the 3290 Information Panel, and the 3287 and 3286 printers. A specific device type is used only when a distinction is required between device types. Information about display terminal usage also refers to the IBM 3138, 3148, and 3158 Display Consoles when used in display mode, unless otherwise noted.

3284. Refers to the IBM 3284 Printer. Information on the 3284 also pertains to the IBM 3286, 3287, 3288, and 3289 printers, unless otherwise noted.

3289. Refers to the IBM 3289 Model 4 Printer.

3310. Refers to the IBM 3310 Direct Access Storage Device.

3330. Refers to the IBM 3330 Disk Storage Device.

3340. Refers to the IBM 3340 Direct Access Storage Device.

3350. Refers to the IBM 3350 Direct Access Storage Device when used in native mode.

3370. Refers to the IBM 3370 Direct Access Storage Device.

3375. Refers to the IBM 3375 Direct Access Storage Device.

3380. Refers to the IBM 3380 Direct Access Storage Device.

3422. Refers to the IBM 3422 Magnetic Tape Subsystem.

3480. Refers to the IBM 3480 Magnetic Tape Subsystem.

370x. Refers to the IBM 3704/3705 Communication Controllers.

3800. Refers to the IBM 3800 Printing Subsystems. A specific device type is used only when a distinction is required between device types.

3850. Refers to the IBM 3850 Mass Storage System.

4

4248. Refers to the IBM 4248 Printer.

9

9313. Refers to the IBM 9313 Direct Access Storage Device.

9332. Refers to the IBM 9332 Direct Access Storage Device, Model 400.

9335. Refers to the IBM 9335 Direct Access Storage Device, Models A01 and B01.

9347. Refers to the IBM 9347 Tape Drive.

Bibliography

Prerequisite Publications

A new user of CMS might refer to the *VM/SP CMS Primer*, SC24-5236, for introductory tutorial information on using CMS.

In addition to the *VM/SP CMS User's Guide*, SC19-6210, prerequisite information is contained in the following publications:

- For information about the terminal that you are using, including procedures for gaining access to the VM/SP system and logging on, see the *Virtual Machine/System Product Terminal Reference*, GC19-6206.
- If you are using an IBM 3767 Communications Terminal, the *IBM 3767 Operator's Guide*, GA18-2000, is a prerequisite.

If you are going to use an IBM Licensed Program compiler under CMS, you should have available the appropriate Licensed Program documentation. These publications are listed in *Virtual Machine/System Product Introduction*, GC19-6200.

Corequisite Publications

The CP commands that are available to you in CP privilege classes A and G, are described in *Virtual Machine/System Product CP General User Command Reference*, SC19-6211.

The *Virtual Machine/System Product System Messages and Codes*, SC19-6204, describes all of the error messages and system responses produced by the CMS commands referenced in this publication. The *Virtual Machine/System Product System Messages Cross Reference*, SC19-6204, contains the cross reference lists formerly contained in the appendix of SC19-6204.

If you are alternating between CMS and other operating systems in virtual machines running under VM/SP, you should consult *Virtual Machine/System Product Running Guest Operating Systems*, GC19-6212.

For information on the CMS functions, CMS macro instructions, and callable services library (CSL) routines refer to the

Virtual Machine/System Product Application Development Reference for CMS, SC24-5284.

For information on the VM/SP System Product Editor refer to the:

Virtual Machine/System Product

System Product Editor Command and Macro Reference, SC24-5221 and
System Product Editor User's Guide, SC24-5220.

For information about the System Product Interpreter, see the:

Virtual Machine/System Product

System Product Interpreter Reference, SC24-5239 and
System Product Interpreter User's Guide, SC24-5238.

For information on EXEC 2 refer to *VM/SP EXEC 2 Reference*, SC24-5219.

Supplemental Publications

For general information about the VM/SP system, see *Virtual Machine/System Product Introduction*, GC19-6200.

Additional descriptions of various CMS functions and commands which are normally used by system support personnel are described in:

Virtual Machine/System Product

Installation Guide, SC24-5237

Operator's Guide, SC19-6202

Planning Guide and Reference, SC19-6201

Application Development Guide for CMS, SC24-5286

Administration, SC24-5285

Connectivity Programming Guide and Reference, SC24-5377

System Facilities for Programming, SC24-5288

VM/SP Diagnosis Guide, LY24-5241

For information on the terms used in VM/SP and for assistance in locating specific information, refer to the *VM/SP Library Guide and Master Index*, GC19-6207.

Information on VM/SP IPCS commands, which are invoked under CMS, is contained in *VM/SP Interactive Problem Control System Guide and Reference*, SC24-5260.

If you plan to use the Remote Spooling Communications Subsystem, see the *RSCS Planning and Installation*, SH24-5057.

Assembler language programmers may find information about the VM/SP assembler in *OS/VS, DOS/VS, and VM/370 Assembler Language*, GC33-4010, and *OS/VS and VM/370 Assembler Programmer's Guide*, GC33-4021.

Details on the CMS CPEREP, a command used to generate output reports from VM/SP error recording records, are contained in:

Virtual Machine/System Product OLTSEP and Error Recording Guide, SC19-6205.

For more details on the operands used with CPEREP, refer to:

OS/VS, DOS/VSE, VM/370 Environmental Recording, Editing, and Printing (EREP) Program, GC28-0772.

For messages issued by CMS CPEREP, see:

OS/VS, DOS/VSE, VM/370 EREP Messages, GC38-1045.

For information on IBM GAM/SP, refer to:

OS/VS Graphic Programming Services (GPS) for IBM 2250 Display Unit and IBM 3250 Graphics Display System, SC27-6971

CMS GAM/SP User's Guide, LC33-0126

Details on the CMS IOCP command are contained in:

Input/Output Configuration Program User's Guide and Reference, GC28-1027.

For information about OS/VS tape label processing, refer to:

OS/VS Tape Labels, GC26-3795.

Ready References for VM/SP Users

There are six publications available as ready reference material when you use VM/SP. They are:

Virtual Machine/System Product

CP General User Command Reference Summary, SX24-5219

CMS Command Reference Summary, SX24-5220

EXEC 2 Language, SX24-5124

Quick Reference, SX20-4400

SP Editor Command Language, SX24-5122

System Product Interpreter Reference Summary, SX20-5126

For VSAM and Access Method Services Users

CMS support of Access Method Services is based on VSE and VSE/VSAM. The control statements that you can use are described in *Using VSE/VSAM Commands and Macros*, SC24-5144. The *VM/SP CMS User's Guide* contains details on how to use this support. Error messages produced by the Access Method Services program, and return codes and reason codes are listed in *VSE/VSAM Messages and Codes*, SC24-5146.

For additional information refer to the *VSE/VSAM Programmer's Reference*, SC24-5145.

For a detailed description of VSE/VSAM macros and macro parameters, refer to the *VSE/Advanced Functions Macro User's Guide*, SC24-5210 and *VSE/Advanced Functions Macro Reference*, SC24-5211. For information on OS/VS VSAM macros, refer to *OS/VS Virtual Storage Access Method (VSAM) Programmer's Guide*, GC26-3838.

For CMS/DOS Users

The CMS ESERV command invokes the VSE ESERV program, and uses, as input, the control statements that you would use in VSE. These control statements are described in *Guide to the DOS/VSE Assembler*, GC33-4024.

Linkage editor control statements, used when invoking the linkage editor under CMS/DOS, are described in *VSE System Control Statements*, SC33-6095, and *MVS/XA Linkage Editor and Loader User's Guide*, GC26-4143.

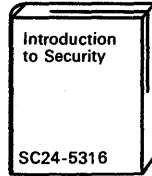
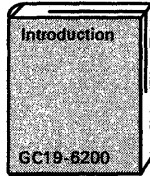
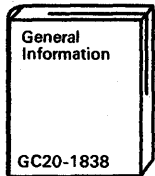
Batch DL/I application programs can be written and tested in the CMS/DOS environment. See *VM/SP CMS User's Guide*, and *DL/I DOS/VS General Information*, GH20-1246, for details.

For information on VSE and CMS/DOS tape label processing, refer to:

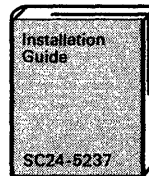
VSE/Advanced Functions Tape Labels, SC24-5212.

VM/SP RELEASE 6 LIBRARY

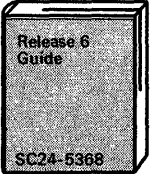
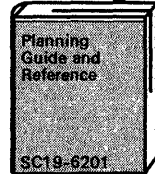
Evaluation



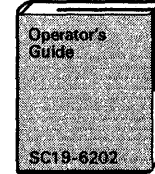
Installation and Service



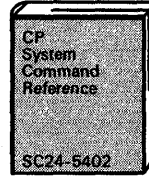
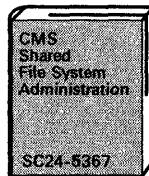
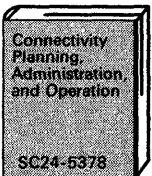
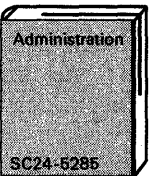
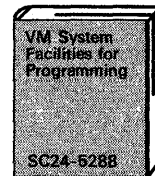
Planning



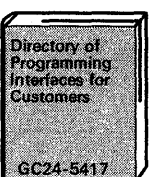
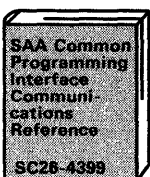
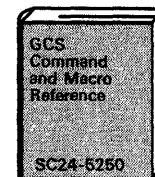
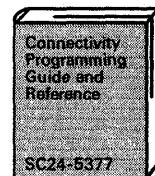
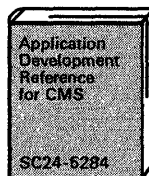
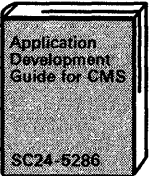
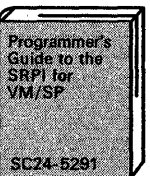
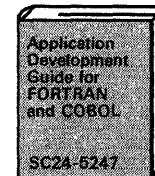
Operation



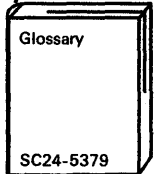
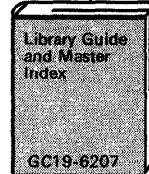
Administration



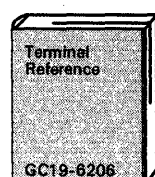
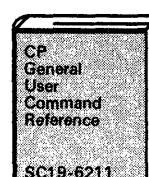
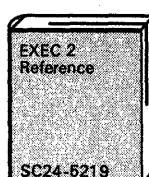
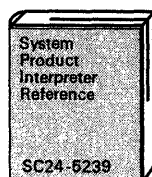
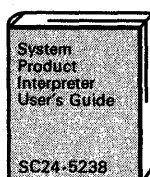
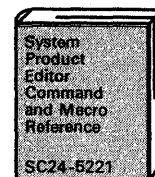
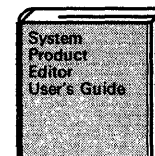
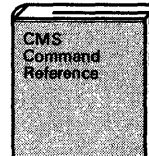
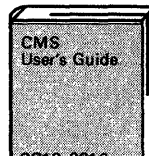
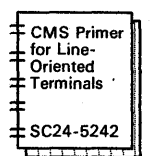
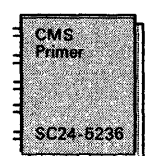
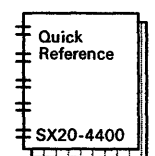
Application Development




Index/Glossary



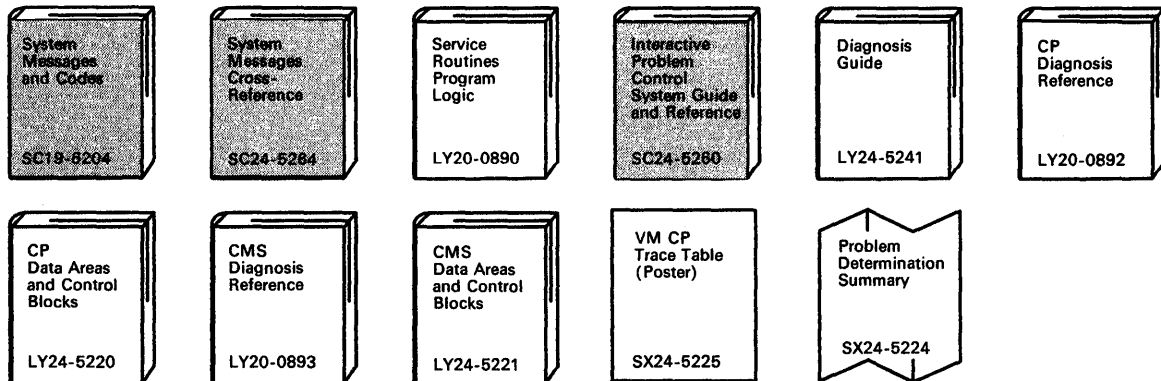
End Use



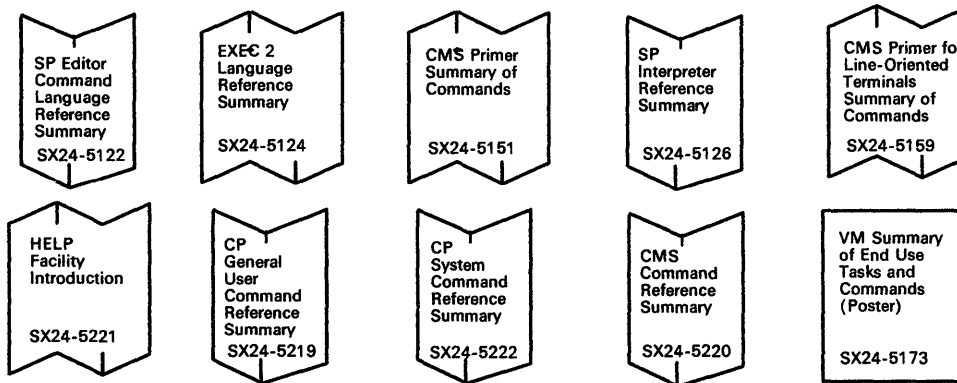
 one copy of each shaded manual received with product tape

VM/SP RELEASE 6 LIBRARY

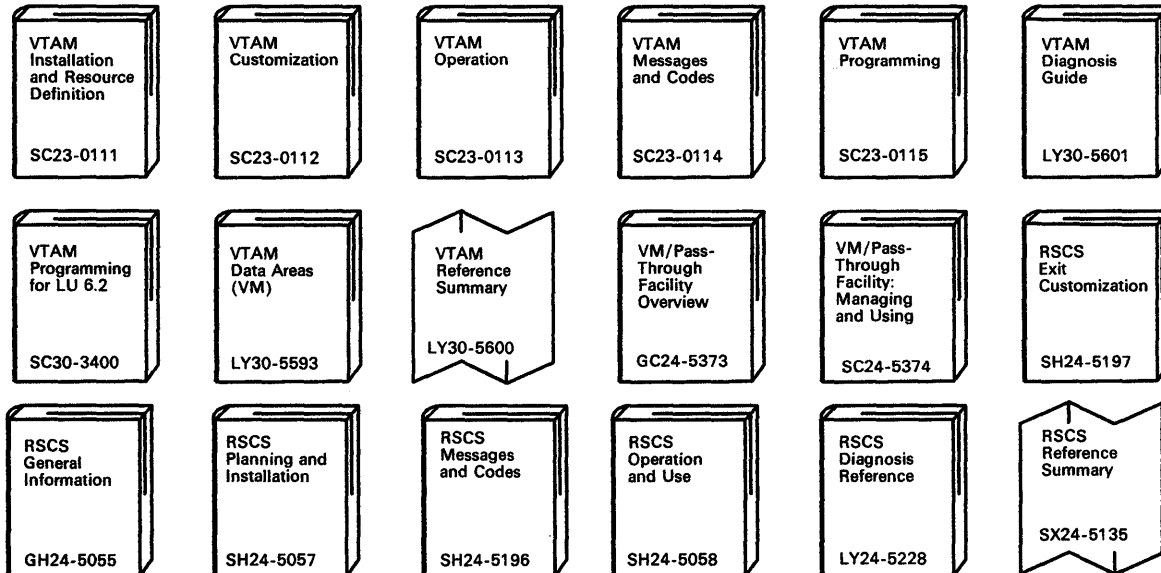
Diagnosis



Reference Summaries



Auxiliary Communication Support



Index

A

- A option of LISTIO command 300
- A-disk accessed after IPLing CMS 29
- ABBREV option
 - of CMS QUERY command 417
 - of CMS SET command 538
 - relationship to SYNONYM command 606
- abbreviations
 - of command names 6, 538, 606
 - querying acceptability of 417
 - setting acceptability of 538
 - used with synonyms 606
- ABENDs
 - effect on DLBL definitions 115
 - effect on FILEDEF definitions 194
 - effect on virtual screen definitions 680
 - effect on window definitions 684
 - encountered by CMSBATCH command 50
- ABNEXIT Macro
 - See VM/SP Application Development Reference for CMS
- abnormal termination (abend)
 - effect on DLBL definitions 115
 - effect on FILEDEF definitions 194
 - effect on virtual screen definitions 680
 - effect on window definitions 684
 - encountered by CMSBATCH command 50
- ACCESS command 27
 - ERASE option 28
 - examples 30
 - first command after IPL 29
 - NODISK option 28
 - NOPROF option 28
 - NOSAVE option 28
 - read-only access 30
 - SAVEONLY option 28
 - shared file 27
 - usage with DEFINE command 31
- access method services (AMS)
 - allocating VSAM space
 - for OS VSAM users 123
 - in CMS/DOS 118
 - determine free space extents for 285
 - invoking in CMS 33
 - LISTING file created by 33
- ACCESSED option of the CMS QUERY command 418
- accessing minidisks with a saved file directory 28, 30
- ACK option
 - of NOTE command 377
 - of SENDFILE command 528
- activating message repository files 563
- ADCON-free module, installing as nucleus extension 385
- ADD option
 - of MACLIB command 331
 - of NOTE command 377
 - of TXTLIB command 629
- address
 - valid 481
- ALARM VSCREEN command 670
 - description 673
 - format 673
 - messages 673
 - operands
 - vname 673
 - responses 673
- alarm, sounding 673
- alias 73
 - definition of 73
 - deleting 75
 - displaying specified alias information via LISTFILE 291
 - moving/relocating 75
- ALIGN option of ASSEMBLE command 40
- alignment of boundaries in assembler program statements 40
- ALIGN2 option of LKED command 303
- ALL
 - operand
 - of CONVERT COMMANDS command 55
 - option
 - of GENMOD command 228
 - of LISTIO command 300
- ALLOC option of LISTFILE command 294
- allocating blocks of formatted minidisk to CMS file using RESERVE command 493
- ALOGIC option of ASSEMBLE command 37
- AMS (access method services)
 - allocating VSAM space
 - for OS VSAM users 123
 - in CMS/DOS 118
 - determine free space extents for 285
 - invoking in CMS 33
 - LISTING file created by 33
- AMSERV
 - command
 - description 33
 - LISTING file 33
 - PRINT option 33
 - TAPIN option 33
 - TAPOUT option 33
- APL character conversion
 - activating in full-screen CMS 539, 728
 - displaying status in CMS 421, 705

- APL option
 - of CMS QUERY command 421, 705
 - of CMS SET command 539, 728
- APPEND option
 - of COPYFILE command 61
 - of FILELIST command 204
 - of LISTFILE command 293
 - of LISTIO command 300
 - of RDRLIST command 459
- applications in full-screen CMS, running 745
- arguments
 - on RUN command 515
 - on START command 597
 - passed to EXEC procedure
 - assigning them to special variables 149
- ASSEMBLE
 - assembler input ddname 41
 - command
 - ALIGN option 40
 - ALOGIC option 37
 - BUFSIZE option 40
 - DECK option 38
 - description 36
 - DISK option 38
 - ESD option 37
 - FLAG option 37
 - LIBMAC option 37
 - LINECOUN option 37
 - LIST option 37
 - listing control options for 37
 - MCALL option 37
 - MLOGIC option 37
 - NOALIGN option 40
 - NOALOGIC option 37
 - NODECK option 38
 - NOESD option 37
 - NOLIBMAC option 38
 - NOLIST option 37
 - NOMCALL option 37
 - NOMLOGIC option 37
 - NONUM option 39
 - NOOBJECT option 39
 - NOPRINT option 38
 - NORENT option 40
 - NORLD option 37
 - NOSTMT option 39
 - NOTERM option 39
 - NOTEST option 39
 - NOXREF option 38
 - NOYFLAG option 40
 - NUMBER option 39
 - OBJECT option 38
 - PRINT option 38
 - RENT option 40
 - RLD option 37
 - STMT option 39
 - SYSPARM option 40
 - TERMINAL option 39
 - TEST option 39
 - ASSEMBLE (*continued*)
 - command (*continued*)
 - WORKSIZE option 41
 - XREF option 38
 - YFLAG option 40
 - filetype
 - created by the TAPPDS command 622
 - used as input to assembler 36
 - assembler
 - conditional assembly statements, listing 37
 - overriding CMS file defaults 41
 - using under CMS 36
 - ASSGN command
 - DEN option 44
 - description 43
 - IGN option 44
 - LOWCASE option 44
 - PRINTER option 43
 - PUNCH option 43
 - READER option 43
 - SYSxxx option 43
 - TAPn option 44
 - TERMINAL option 44
 - TRTCH option 44
 - UA option 44
 - UPCASE option 44
 - 7TRACK option 44
 - 9TRACK option 44
 - assigning logical segments in physical segments 518
 - assignment
 - logical unit, listing 300
 - system and programmer, unassigning 481
 - attention interruption, causing 11
 - ATTN Function
 - See VM/SP Application Development Reference for CMS
 - AUTO option
 - of INCLUDE command 273
 - of LOAD command 309
 - automatic
 - read function, setting 540
 - AUTOREAD option
 - of CMS QUERY command 424
 - of CMS SET command 540
 - auxiliary directory, creating 225
 - AUXPROC option of FILEDEF command 195
- B**
 - B border command 779
 - base file 73
 - definition of 73
 - displaying specified base file information via LISTFILE 291
 - BCD characters, converting to EBCDIC 62
 - BDAM files, specifying in CMS 192
 - blanks
 - as delimiters 2

- BLIP option
 - of CMS SET command 541
 - BLKSIZE option
 - of FILEDEF command 191
 - of FORMAT command 221
 - of TAPE command 612
 - BLOCK option of FILEDEF command 191
 - blocksize, specifying with FILEDEF command 191
 - BLP operand of FILEDEF command 198
 - border commands
 - B (backward) 779
 - C (clear) 779
 - D (drop) 779
 - description 778
 - entering on border corners 778
 - F (forward) 780
 - H (hide) 780
 - L (left) 780
 - M (move) 781
 - messages 779
 - N (minimize) 781
 - O (restore) 781
 - R (right) 782
 - S (size) 782
 - usage notes 778
 - X (maximize) 783
 - BORDER option
 - of CMS QUERY command 706
 - of CMS SET command 729
 - border, window
 - changing 729
 - defining 729
 - displaying attributes 706
 - displaying status 706
 - entering border commands from corners 778
 - boundary alignment of statements in assembler program 40
 - BOX (.BX) format word 798
 - BRIEF HELP
 - described 258
 - obtaining using HELP command 255
 - obtaining using MOREHELP command 343
 - BRKKEY setting changed for full-screen CMS 740
 - BRKKEY setting, default changed 740
 - BSF tape control function 611
 - BSR tape control function 611
 - buffer size
 - controlling for assembler 40
 - for VSAM programs 116
 - BUFSIZE option of ASSEMBLE command 40
 - BUFSP option of DLBL command 116
- C**
- C border command 779
 - callable services library (CSL) routines iv
 - binding a CSL routine to a fixed location 503
 - displaying 85
 - callable services library (CSL) routines (*continued*)
 - displaying information about loaded and bound CSL routines 507
 - dropping 501
 - loading a CSL routine 503
 - searching for a CSL routine 503
 - verifying the existence of CSL routines 511
 - calling the parsing facility from an exec 395
 - CANCEL option of NOTE command 377
 - canceling immediate commands 269
 - card deck, reading into virtual card reader 468
 - CARD option of EXECIO command 157
 - CAT option
 - of DLBL command 116
 - example of use 125
 - use in CMS/DOS 121
 - catalog, VSAM, verifying structure of 47
 - CATCHECK command described 47
 - CC option
 - of EXECIO command 158
 - of NOTE command 377
 - of PRINT command 404
 - CD option of DSERV command 135
 - CHANGE option
 - of DLBL command 115
 - of FILEDEF command 191
 - of LABELDEF command 279
 - changing location of windows 699
 - changing number of lines and columns in a window 766
 - changing the current language 563
 - changing virtual screen attributes 757
 - changing window attributes 759
 - character
 - defining logical line end in CMS 568, 748
 - displaying CMS logical line end 439, 710
 - nondisplayable, defining character used in place of 572, 752
 - nondisplayable, displaying character used in place of 442, 711
 - sets used in CMS 3
 - strings
 - copying 68
 - valid in CMS command lines 3
 - CHARMODE option
 - of CMS QUERY command 706
 - of CMS SET command 732
 - CHARS option of SETPRT command 590
 - CLEAR key in full-screen CMS 698, 745
 - CLEAR operand of IMMCMD command 269
 - CLEAR option
 - of DLBL command 115
 - of FILEDEF command 188
 - of INCLUDE command 271
 - of LABELDEF command 277
 - of LOAD command 309
 - of SYNONYM command 606

CLEAR VSCREEN command 670
 description 674
 format 674
 messages 674
 operands
 vname 674
 usage notes 674

CLEAR WINDOW command 670
 description 675
 format 675
 messages 675
 operands
 wname 675
 usage notes 675

clearing a stack 100
 clearing a window 675
 closing files 219
 CMDCALL command described 49
 CMS batch facility
 CMS editor in migration mode 138
 CMS EXEC file
 appending information to 293
 creating 293
 format 295
 CMS files
 See files
 CMS Functions
 See VM/SP Application Development Reference for CMS
 CMS immediate commands
 See immediate commands
 CMS LOADLIBs
 compressing with LOADLIB command 324
 copying with LOADLIB command 324
 creating with LKED command 302
 executing a load module from 394
 listing with LOADLIB command 324
 CMS Macro Instructions
 See VM/SP Application Development Reference for CMS
 CMS (Conversational Monitor System)
 accessing with no virtual disks attached to virtual machine 28
 basic description 1
 batch facility
 See CMS batch facility
 command language, basic description 1
 commands
 See CMS (Conversational Monitor System) commands
 files
 See files
 loader
 See loader
 option of DLBL command 115
 CMS (Conversational Monitor System) commands
 ACCESS 27
 ALARM VSCREEN 673

CMS (Conversational Monitor System) commands

(continued)

ALIALIST 785
 AMSERV 33
 ASSEMBLE 36
 ASSGN 43
 AUTHLIST 785
 border commands 778
 CLEAR VSCREEN 674
 CLEAR WINDOW 675
 CMDCALL 49
 CMSBATCH 50
 CMSSERV 51
 COMPARE 53
 CONVERT COMMANDS 55
 CONWAIT 59
 COPYFILE 60
 CP 72
 CREATE ALIAS 73
 CREATE DIRECTORY 77
 CREATE LOCK 79
 CREATE NAMEDEF 83
 CSLLIST 85
 CURSOR VSCREEN 676
 DEFAULTS 93
 DEFINE VSCREEN 679
 DEFINE WINDOW 683
 DELETE LOCK 96
 DELETE NAMEDEF 99
 DELETE VSCREEN 686
 DELETE WINDOW 687
 DESBUF 100
 DIRLIST 101
 DISCARD 785
 DISK 108
 DLBL 114
 DOSLIB 128
 DOSLKED 130
 DROPBUF 134
 DSERV 135
 EDIT 138
 entering 2
 entering by synonym 606
 ERASE 141
 ESERV 147
 ESTATE 600
 ESTATEW 600
 EXEC 149
 EXECDROP 153
 EXECIO 155
 EXECLOAD 171
 EXECMAP 173
 EXECOS 176
 EXECSTAT 178
 EXECUPDT 180
 EXECUTE 785
 execution characteristics 10
 FETCH 183

CMS (Conversational Monitor System) commands

(continued)

FILEDEF 186
 FILELIST 203
 FINIS 219
 FORMAT 220
 GENDIRT 225
 GENMOD 226
 GENMSG 233
 GET VSCREEN 690
 GLOBAL 237
 GLOBALV 240
 GRANT AUTHORITY 251
 halting execution 666
 HELP 255
 HELPCONV 265
 HIDE WINDOW 692
 IDENTIFY 267
 IMMCMD 269
 INCLUDE 271
 LABELDEF 277
 LISTDIR 282
 LISTDS 285
 LISTFILE 291
 LISTIO 300
 LKED 302
 LOAD 308
 LOADLIB 324
 LOADMOD 328
 MACLIB 331
 MACLIST 335
 MAKEBUF 341
 MAXIMIZE WINDOW 694
 MINIMIZE WINDOW 696
 MODMAP 342
 MOREHELP 343
 MOVEFILE 345
 NAMEFIND 349
 NAMES 358
 NETDATA 364
 NOTE 376
 NUCXDROP 384
 NUCXLOAD 385
 NUCXMAP 389
 OPTION 392
 OSRUN 394
 PARSECMD 395
 PEEK 398
 POP WINDOW 697
 POSITION WINDOW 699
 PRINT 403
 PROGMAP 407
 PSERV 410
 PUNCH 412
 PUT SCREEN 700
 PUT VSCREEN 702
 QUERY 415, 704
 QUERY LOADAREA 439

CMS (Conversational Monitor System) commands

(continued)

QUERY STORECLR 447
 RDR 454
 RDRLIST 459
 READCARD 466
 RECEIVE 472
 REFRESH 719
 RELEASE 481
 RELOCATE 484
 RENAME 488
 RESERVE 493
 RESTORE WINDOW 720
 REVOKE AUTHORITY 496
 ROUTE 721
 RSERV 499
 RTNDROP 501
 RTNLOAD 503
 RTNMAP 507
 RTNSTATE 511
 RUN 514
 SCROLL 724
 search order 9
 SEGMENT 517
 SENDFILE 527
 SENTRIES 536
 SET 537, 727
 SETKEY 589
 SETPRT 590
 SHOW WINDOW 764
 SIZE WINDOW 766
 SORT 593
 SSERV 595
 START 597
 STATE 600
 STATEW 600
 summary of CMS commands 11
 summary of CMS commands in other
 publications 20, 21, 22, 23
 SVCTRACE 602
 SYNONYM 606
 TAPE 610
 TAPEMAC 619
 TAPPDS 622
 TELL 627
 transient area 10
 TXTLIB 629
 TYPE 633
 UPDATE 636
 user area 10
 VALIDATE 649
 WAITREAD VSCREEN 767
 WAITT VSCREEN 770
 WRITE VSCREEN 772
 XEDIT 651
 XMITMSG 658
 CMSAMS saved system name 582

CMSBAM saved system name 582
 CMSBATCH command
 description 50
 recursive abends encountered by 50
 CMSDOS saved system name 582
 CMSLEVEL option, of CMS QUERY command 424
 CMSLIB assembler macro library ddname 41
 CMSPF keys
 canceling 735
 defining to execute a command 734
 displaying definitions 707
 RETRIEVE function 735
 using NOWRITE option 735
 CMSPF option
 of CMS QUERY command 707
 of CMS SET command 734
 CMSSERV command
 description 51
 format 51
 messages 52
 usage notes 51
 CMSTYPE option
 of CMS QUERY command 424
 of CMS SET command 542
 CMSUT1 file
 created by DISK LOAD command 110
 created by READCARD command 468
 created by TAPE LOAD command 615
 created by TAPPDS command 622
 CMSVSAM saved system name 582
 COBOL
 compilers
 querying options in effect for 443
 specifying options for in CMS/DOS 392
 COL option
 of COMPARE command 53
 of TYPE command 633
 columns
 comparing files by 53
 displaying particular
 with TYPE command 633
 of data, copying 68
 specifying
 for copy operations 68
 columns, changing number in a window 766
 COL1 option of TAPPDS command 623
 COMDIR option of CMS SET command 543
 commands
 abbreviating 6
 defaults shown by underscore in command format
 box 7
 entering 2
 entering from the WM window 697
 environment
 CMS 1
 CP 1
 execution, halting 666
 keyboard differences in entering 12
 commands (*continued*)
 language, CMS 1
 modules, creating 226
 operands 2
 options 2
 stacking in terminal input buffer 12
 truncating 6
 when to enter 11
 COMMANDS ASSEMBLE utility file 56
 COMMANDS CMSUT1 utility file 56
 COMMANDS CMSUT2 utility file 56
 COMMANDS TEXT utility file 56
 commands you can enter in the WM window 697
 COMMENT (.CM) format word 800
 COMMENTS option of EXECUPDT command 181
 Communications
 NETDATA command 364
 NOTE command 364
 RECEIVE command 364
 SENDFILE command 364
 COMP
 of DOSLIB command 128
 option
 of FETCH command 183
 of MACLIB command 331
 COMPARE command
 COL option 53
 description 53
 comparing files 53
 compilers used under CMS 1
 compiling a message repository file 233
 components
 of VM/SP 1
 COMPRESS option
 of EXECUPDT command 181
 of LOADLIB command 324
 COMPSWT Macro
 See VM/SP Application Development Reference for
 CMS
 CONCAT option of FILEDEF command 192
 CONDITIONAL SECTION (.CS) format word 801
 connecting a window to a virtual screen 692, 764
 console
 read, controlling after CMS command
 execution 540
 stack
 clearing 100
 terminal input buffer
 clearing 100
 constants, altering
 with LOAD command 321
 continuation character
 on COPYFILE specification list 68
 on COPYFILE translation list 70
 Control Program (CP)
 See also CP (Control Program)
 basic description 1
 commands
 See CP (Control Program) commands

Control Program (CP) commands
 description 72
 executing
 in CMS command environment 72
 in EXEC procedure 72
 in jobs for CMS batch facility 72
 implied 559
 when to use 72
 control statements
 for access method services 33
 for UPDATE command 638
 conventions, notational 6
 Conversational Monitor System (CMS)
 accessing with no virtual disks attached to virtual
 machine 28
 basic description 1
 batch facility
 See CMS batch facility
 command language, basic description 1
 commands
 See CMS (Conversational Monitor System)
 commands
 files
 See files
 loader
 See loader
 Conversational Monitor System (CMS) commands
 ACCESS 27
 ALARM VSCREEN 673
 ALIALIST 785
 AMSERV 33
 ASSEMBLE 36
 ASSGN 43
 AUTHLIST 785
 border commands 778
 CLEAR VSCREEN 674
 CLEAR WINDOW 675
 CMDCALL 49
 CMSBATCH 50
 CMSSERV 51
 COMPARE 53
 CONVERT COMMANDS 55
 CONWAIT 59
 COPYFILE 60
 CP 72
 CREATE ALIAS 73
 CREATE DIRECTORY 77
 CREATE LOCK 79
 CREATE NAMEDEF 83
 CSLLIST 85
 CURSOR VSCREEN 676
 DEFAULTS 93
 DEFINE VSCREEN 679
 DEFINE WINDOW 683
 DELETE LOCK 96
 DELETE NAMEDEF 99
 DELETE VSCREEN 686
 DELETE WINDOW 687

Conversational Monitor System (CMS) commands
 (continued)
 DESBUF 100
 DIRLIST 101
 DISCARD 785
 DISK 108
 DLBL 114
 DOSLIB 128
 DOSLKED 130
 DROPBUF 134
 DSERV 135
 EDIT 138
 entering 2
 entering by synonym 606
 ERASE 141
 ESERV 147
 ESTATE 600
 ESTATEW 600
 EXEC 149
 EXECDROP 153
 EXECIO 155
 EXECLOAD 171
 EXECMAP 173
 EXECOS 176
 EXECSTAT 178
 EXECUPDT 180
 EXECUTE 785
 execution characteristics 10
 FETCH 183
 FILEDEF 186
 FILELIST 203
 FINIS 219
 FORMAT 220
 GENDIRT 225
 GENMOD 226
 GENMSG 233
 GET VSCREEN 690
 GLOBAL 237
 GLOBALV 240
 GRANT AUTHORITY 251
 halting execution 666
 HELP 255
 HELPCONV 265
 HIDE WINDOW 692
 IDENTIFY 267
 IMMCMD 269
 INCLUDE 271
 LABELDEF 277
 LISTDIR 282
 LISTDS 285
 LISTFILE 291
 LISTIO 300
 LKED 302
 LOAD 308
 LOADLIB 324
 LOADMOD 328
 MACLIB 331
 MACLIST 335

Conversational Monitor System (CMS) commands

(continued)

MAKEBUF 341
 MAXIMIZE WINDOW 694
 MINIMIZE WINDOW 696
 MODMAP 342
 MOREHELP 343
 MOVEFILE 345
 NAMEFIND 349
 NAMES 358
 NETDATA 364
 NOTE 376
 NUCXDROP 384
 NUCXLOAD 385
 NUCXMAP 389
 OPTION 392
 OSRUN 394
 PARSECMD 395
 PEEK 398
 POP WINDOW 697
 POSITION WINDOW 699
 PRINT 403
 PROGMAP 407
 PSERV 410
 PUNCH 412
 PUT SCREEN 700
 PUT VSCREEN 702
 QUERY 415, 704
 QUERY LOADAREA 439
 QUERY STORECLR 447
 RDR 454
 RDRLIST 459
 READCARD 466
 RECEIVE 472
 REFRESH 719
 RELEASE 481
 RELOCATE 484
 RENAME 488
 RESERVE 493
 RESTORE WINDOW 720
 REVOKE AUTHORITY 496
 ROUTE 721
 RSERV 499
 RTNDROP 501
 RTNLOAD 503
 RTNMAP 507
 RTNSTATE 511
 RUN 514
 SCROLL 724
 search order 9
 SEGMENT 517
 SENDFILE 527
 SENTRIES 536
 SET 537, 727
 SETKEY 589
 SETPRT 590
 SHOW WINDOW 764
 SIZE WINDOW 766

Conversational Monitor System (CMS) commands

(continued)

SORT 593
 SSERV 595
 START 597
 STATE 600
 STATEW 600
 summary of CMS commands 11
 summary of CMS commands in other
 publications 20, 21, 22, 23
 SVCTRACE 602
 SYNONYM 606
 TAPE 610
 TAPEMAC 619
 TAPPDS 622
 TELL 627
 transient area 10
 TXTLIB 629
 TYPE 633
 UPDATE 636
 user area 10
 VALIDATE 649
 WAITREAD VSCREEN 767
 WAIT VSCREEN 770
 WRITE VSCREEN 772
 XEDIT 651
 XMITMSG 658
 CONVERT COMMANDS command
 description 55
 examples 57
 format 55
 messages 57
 operands
 ALL 55
 CHECK 55
 FIFO 56
 fn ft fm 55
 LIFO 56
 OUTMODE 56
 STACK 56
 SYSTEM 55
 USER 55
 usage notes 57
 converting DLCS files
 checking for DLCS coding errors 55
 converting for the parsing facility 55
 example 57
 from an EXEC 57
 from XEDIT 57
 output files 56
 utility files 56
 converting message repository files
 converting into internal form 233
 example 235
 output files 234
 CONWAIT command
 description 59
 using 59

COPIES option of SETPRT command 590
 COPY
 filetype
 adding to MACLIBs 332
 created by SSERV command 595
 option of LOADLIB command 324
 COPYFILE command 60
 APPEND option 61
 character translations 69
 EBCDIC option 62
 examples 64
 FILL option 62
 FOR option 61
 FRLABEL option 61
 FROM option 61
 incompatible options 63
 LOWCASE option 62
 LRECL option 62
 NEWDATE option 61
 NEWFILE option 61
 NOPROMPT option 61
 NOSPECS option 61
 NOTRUNC option 62
 NOTYPE option 61
 OLDDATE option 61
 OVLY option 61
 PACK option 62
 PROMPT option 61
 RECFM option 62
 REPLACE option 61
 SINGLE option 63
 specification list 68
 SPECS option 61
 TOLABEL option 61
 TRANS option 63
 TRUNC option 62
 TYPE option 61
 UNPACK option 62
 UPCASE option 62
 usage 64
 copying books from VSE source statement
 libraries 595
 copying the physical screen to a CMS file 700
 copying virtual screen data to a CMS file 702
 copying.copied files 60
 COPYnr option of SETPRT command 590
 CP operand of EXECIO command 157
 CP (Control Program)
 basic description 1
 commands
 See CP (Control Program) commands
 CP (Control Program) commands
 description 72
 executing
 in CMS command environment 72
 in EXEC procedure 72
 in jobs for CMS batch facility 72
 implied 559

CP (Control Program) commands (*continued*)
 when to use 72
 CRDTE operand of LABEL command 278
 CREATE ALIAS command 73
 CREATE DIRECTORY command 77
 CREATE LOCK command 79
 CREATE NAMEDEF command 83
 creating a program stack buffer via MAKEBUF 341
 creating a segment space 525
 creating a virtual screen 679
 creating a window 683
 CSECTs duplicated for LOAD command 310
 CSL (callable services library) routines iv
 CSLLIST command 85
 CTL
 option
 of EXECUPDT command 180
 of UPDATE command 637
 of XEDIT command 653
 cursor
 displaying location in virtual screen 708
 displaying location on physical screen 707
 positioning 676
 CURSOR option
 of CMS QUERY command 707
 CURSOR VSCREEN command 670
 description 676
 format 676
 messages 678
 operands
 col 676
 DATA 676
 line 676
 RESERVED 676
 vname 676
 responses 678
 usage notes 677
 cursor, positioning 676

D

D border command 779
 D-disk accessed after IPLing CMS 29
 data transmission, handling remote 579, 753
 data transmission, querying remote 444, 711
 DATE option of LISTFILE command 294
 DD (data definition), simulating in CMS 186
 ddnames
 defining
 with DLBL command 114
 with FILEDEF command 186
 entering tape ddnames for AMSERV 34
 for DLBL command, restrictions for OS users 116
 relating to CMS file 186
 to identify VSAM catalogs (CMS/DOS) 121
 to identify VSAM catalogs (OS VSAM users) 125
 used by assembler 41

DECK option
 of ASSEMBLE command 38
 of OPTION command 392
DEFAULTS command 93
 LIST option 93
 SET option 93
 valid CMS command options 93
DEFINE VSCREEN command 670
 description 679
 examples 681
 format 679
 messages 682
 operands
 color 680
 cols 679
 exthi 680
 HIGH 680
 lines 679
 NOHIGH 680
 NOPROTECT 680
 NOTYPE 679
 PROTECT 680
 psset 680
 rbot 679
 rtop 679
 SYSTEM 680
 TYPE 679
 USER 680
 vname 679
 usage notes 680, 684
DEFINE WINDOW command 670
 description 683
 format 683
 messages 685
 operands
 BORDER 683
 cols 683
 FIXED 683
 lines 683
 NOBORDER 683
 NOPOP 684
 NOTOP 684
 POP 684
 pscol 683
 psline 683
 SYSTEM 684
 TOP 684
 USER 684
 VARIABLE 683
 wname 683
 defining a virtual screen 679
 defining a window 683
Definition Language for Command Syntax (DLCS)
 See DLCS (Definition Language for Command Syntax)
DEL option
 of DOSLIB command 128
 of MACLIB command 331
DEL option (continued)
 of TXTLIB command 629
DELETE
 control statement, for UPDATE command 640
DELETE LOCK command 96
DELETE NAMEDEF command 99
DELETE VSCREEN command 670
 description 686
 format 686
 messages 686
 operands
 vname 686
 usage notes 686
DELETE WINDOW command 670
 description 687
 format 687
 messages 687
 operands
 oper 687
 usage notes 687
 deleting a virtual screen definition 686
 deleting a window definition 687
 deleting, program stack buffer 134
DEN option
 of ASSGN command 44
 of FILEDEF command 193
 of TAPE command 613
DESBUF command described 100
DET option in RELEASE command 481
 detaching a disk from virtual machine configuration 481
DETAIL HELP
 described 258
 obtaining using HELP command 255
 obtaining using MOREHELP command 343
DIAGNOSE X'64' and the SEGMENT command 517
 directing messages 721
directories
 accessing 27
 block size 426
 CMS auxiliary 225
 CMS file, writing to disk 481
 creating 77
 creating a subdirectory 77
 determining available space 427
 determining if accessed 427
 determining type of access authority 427
 displayed as a dash via QUERY command with DISK option 426
 displaying a directory structure 101
 displaying specified directory information via LISTFILE 291
 displaying status 425
 erasing files from 141
 freeing an accessed directory 481
 identifying directories via STAT option of QUERY command 426
 listing a directory structure 282

directories (*continued*)

- locking 77
- moving directory structures 484
- naming 4
- of VSE libraries, sorting 136
- renaming 488
- represented by file mode letter 425
- restrictions on moving 484

directory accessed as file mode A 29

DIRID operand

- definition 4

DIRLIST command 101

DISK

- command
 - description 108
 - DUMP operand 108
 - LOAD operand 108
- FULLPROMPT option 109
- MINPROMPT option 109
- NOPROMPT option 109
- NOREPLACE option 109
- option
 - interactive use with FILEDEF command 197
 - of ASSEMBLE command 38
 - of CMS QUERY command 425
 - of DOSLIB command 128
 - of DOSLKED command 131
 - of DSERV command 136
 - of FILEDEF command 188
 - of LKED command 303
 - of LOADLIB command 325
 - of MACLIB command 332
 - of PSERV command 410
 - of RSERV command 499
 - of SSERV command 595
 - of TAPE command 612
 - of TXTLIB command 630
 - of UPDATE command 637
 - use with FILEDEF command 196
- REPLACE option 109
- verifying access 649

Disk Operating System (DOS)

- disks, accessing 31
- files
 - listing information 285
 - specifying FILEDEF options for 194

DISKID Function

- See* VM/SP Application Development Reference for CMS

DISKR option of EXECIO command 157

disks

- accessing 27
- detaching 481
- determining
 - if disk is full 425
 - read/write status of 425
- determining available space 427
- device type 426

disks (*continued*)

- displaying status 425
- erasing files from 141
- files
 - See* files
- formatting 220
- read/write, sharing 31
- releasing
 - effect on logical unit assignments in
 - CMS/DOS 45
 - in CMS/DOS 482
 - when DLBL definitions are active 123
 - storage capacity, displaying status 425
- DISKW option of EXECIO command 157
- DISP option of FILEDEF command 192
- display
 - message text 256
 - of CMS QUERY command 708
- displaying a variable size window 697
- displaying a window 684, 764
- displaying characteristics of physical screen 708
- displaying information about a loaded CSL routine 507
- displaying the WM window 697

DLBL

- command
 - CAT option 116
 - CHANGE option 115
 - CLEAR option 115
 - CMS option 115
 - ddname restrictions (OS users) 122
 - description 114
 - displaying volumes on which all multivolume data sets reside 121
 - displaying VSAM data set extents 119
 - DSN option 115
 - DUMMY option 115
 - entering OS data set names 115
 - entering the SYSxxx operand 117
 - entering VSE file ID 115
 - establishing file definitions for STATE command 601
 - EXTENT option 116
 - MULT option 116
 - NOCHANGE option 116
 - PERM option 115
 - SYSxxx option 115
 - to identify files for AMSERV 34
 - VSAM option 116
 - when to use (OS users) 122
- definitions
 - cleared by ESERV command 147
 - clearing 115
 - displaying 427
 - option of CMS QUERY command 427

DLCS (Definition Language for Command Syntax)

- checking for DLCS coding errors 55
- converting a DLCS file 55

DLCS (Definition Language for Command Syntax)
(continued)
 converting files from an EXEC 57
 converting from XEDIT 57
 displaying contents of synonym and translation tables 449
 enabling user DLCS definitions 563
 example 57
 output files from conversion 56
 setting translations and synonyms 586
 specifying use of translation tables 586
 utility files 56

DOS option
 of CMS QUERY command 428
 of CMS SET command 545
 of GENMOD command 227

DOS (Disk Operating System)
 disks, accessing 31
 files
 listing information 285
 specifying FILEDEF options for 194

DOSLIB
 command
 COMP option 128
 DEL option 128
 description 128
 DISK option 128
 MAP option 128
 PRINT option 128
 TERM option 128

files
 adding phases to 131
 fetching phases from 183
 identifying for fetching 237
 listing information about members 128
 output file mode 130
 size considerations 129
 space considerations 131
 which DOSLIBs will be searched 429

option
 of CMS QUERY command 429
 of GLOBAL command 237

DOSLKED command
 description 130
 DISK option 131
 PRINT option 131
 TERM option 131

DOSLNCNT option
 of CMS QUERY command 429
 of CMS SET command 547

DOSLNK filetype
 CMS/DOS linkage editor input 130
 creating 131

DOSPART option
 of CMS QUERY command 429
 of CMS SET command 548

DROP WINDOW command 670
 description 688

DROP WINDOW command (continued)
 format 688
 messages 688
 operands
 n 688
 WM 688
 wname 688
 * (asterisk) 688
 = (equal sign) 688
 usage notes 688

DROPBUF command
 description 134
 using 134

dropping a window 688
 dropping execs and editor macros from storage 153
 dropping the WM window 688

DSERV command
 ALL operand 135
 CD operand 135
 description 135
 DISK option 136
 PD operand 135
 PRINT option 136
 RD operand 135
 SD operand 135
 SORT option 136
 TD operand 135
 TERM option 136

DSN option of DLBL command 115
 DSORG option of FILEDEF command 192

DUMMY option
 of DLBL command
 restrictions for OS VSAM user 122
 using in CMS/DOS 117
 of FILEDEF command 188

DUMP
 option
 of DISK command 108
 of OPTION command 392
 of TAPE command 610

DUP option
 of INCLUDE command 273
 of LOAD command 310

DVOL1 operand of TAPE command 612

E

EBCDIC
 display file in 633
 of COPYFILE command 62

EDIT
 command
 description 138
 LRECL option 139
 NODISP option 139
 subcommands
 See EDIT subcommands

EDIT EXEC S2 suppression of execution 139
 EDIT subcommands
 editor
 CMS
 migration mode 138
 macros
 issuing messages 661
 System Product Editor
 environment, issuing CP and CMS commands
 from 654
 invoking 651
 using 654
 emptying a stack 100
 EMSG option of EXECIO command 158, 167
 enabling user DLCS definitions 563
 END option of TAPPDS command 624
 ENDCMD operand of NUCXLOAD command 386
 Enhanced Connectivity Facilities on VM/SP
 starting communications 51
 entering full-screen CMS 739
 ENTRY loader control statement 318
 entry points
 determined by loader 316
 displayed with FETCH command 183
 specifying
 with ENTRY statement 318
 with GENMOD command 227
 EOF option of TAPE command 613
 EOT option of TAPE command 613
 ERASE
 command
 NOTYPE option 142
 TYPE option 142
 option
 of ACCESS command 28
 ERASE command 141
 erasing a defective section of tape 611
 erasing data in a virtual screen 674
 ERG tape control function 611
 ERRS option of OPTION command 393
 escape character
 default 557
 displaying 434
 setting 557
 when not required 557
 ESD option of ASSEMBLE command 37
 ESERV command described 147
 ESTATE command described 600
 ESTATEW command described 600
 ETMODE option of EXECUPDT command 181
 EXCLUDE SYSIN control statement 325
 EXDTE operand of LABELDEF command 278
 EXEC
 command
 description 149
 implied 559
 files
 CMS EXEC file created by LISTFILE
 command 293

EXEC (*continued*)
 files (*continued*)
 executing with the RUN command 514
 using IMMCMD from 269
 \$LISTIO EXEC created by LISTIO
 command 300
 filetype
 record format 149
 option
 of LISTFILE command 293
 of LISTIO command 300
 procedures
 defining synonyms for 606
 ESERV 147
 executing 149
 reading from and writing to windows 768
 RUN EXEC 514
 using windows 768
 using WRITE VSCREEN 774
 special variables
 variables for PARSECMD 396
 variables for WAITREAD VSCREEN 767
 variables returned
 code.n 396
 code.n values 396
 token.n 396
 WAITREAD.n 767
 EXEC 2 procedures, executing 149
 EXECDROP command
 description 153
 EXECIO command
 CARD option 157
 CC option 158
 CP option 157
 description 155
 DISKR option 157
 DISKW option 157
 EMSG option 158
 machine code 158, 166
 PRINT option 158
 PUNCH option 157
 STEM option 160
 VAR option 160
 EXECLOAD command
 description 171
 EXECMAP command
 description 173
 FIFO option 174
 LIFO option 174
 EXECOS command
 description 176
 execs and editor macros in storage
 controlling system searching of CMS installation
 saved segment 561
 discontinue use of the CMS installation saved
 segment 153
 listing 173
 querying status of CMS installation saved
 segment 435

execs and editor macros in storage (*continued*)
 removing 153
 EXECSTAT command
 description 178
 EXECTRAC option
 of CMS QUERY command 430
 of CMS SET command 550
 EXECUPDT command
 COMMENTS option 181
 COMPRESS option 181
 CTL option 180
 description 180
 ETMODE option 181
 HISTORY option 180
 NOCOMMENTS option 181
 NOCOMPRESS option 181
 NOHISTORY option 180
 NOSID option 181
 NOUPDATE option 181
 SID option 181
 used with System Product Interpreter source
 programs 180
 using UPDATE options with 181
 execution characteristics of CMS commands 10
 exiting the WM window 688
 expanding a window size to the physical screen
 size 694
 extensions
 read-only
 accessing 27
 editing files on 138
 releasing 482
 EXTENT option
 of DLBL command 116
 of DLBL for CMS/DOS users 117
 of LISTDS command 286

F

F border command 780
 FCB option of SETPRT command 590
 FETCH command
 COMP option 183
 description 183
 ORIGIN option 183
 START option 183
 FID operand of LABELDEF command 277
 field definition character 768
 FIFO (first-in/first-out) operand
 operand
 of CONVERT COMMANDS command 56
 option
 of IDENTIFY command 267
 of NAMEFIND command 350
 of NUCXMAP command 390
 of RDR command 454
 file directory, accessing with a saved copy 28, 30

file ID in command syntax 7
 file mode S accessed after IPLing CMS 28, 29
 FILE option of NAMEFIND command 350
 FILEDEF
 command
 ALT option 194
 AUXPROC option 195
 BLKSIZE option 191
 BLOCK option 191
 BLP operand 198
 CHANGE option 191
 CLEAR option 188
 CONCAT option 192
 default FILEDEF commands issued by
 assembler 41
 definitions for MOVEFILE command 345
 DEN option 193
 description 186
 DISK option 188
 DISP MOD option 192
 DSORG option 192
 DUMMY option 188
 establishing file definitions for STATE
 command 600
 examples 196
 GRAF option 188
 KEYLEN option 191
 LABOFF operand 198
 LEAVE option 193
 LIMCT option 192
 LOWCASE option 193
 LRECL option 191
 MEMBER option 192
 NL operand 198
 NOCHANGE option 191
 NOEOV option 193
 NSL operand 198
 OPTCD option 192
 PERM option 191
 positioning read/write pointer 192
 PRINTER option 187
 PUNCH option 188
 READER option 188
 RECFM option 191
 SL operand 198
 SUL operand 198
 SYSPARM option 199
 TAPn option 188
 TERMINAL option 187
 TRTCH option 193
 UPCASE option 193
 VOLID operand 198
 when to use in CMS/DOS 118
 when to use (OS users) 122
 XTENT option 192
 18TRACK option 193
 7TRACK option 193
 9TRACK option 193

FILEDEF (continued)

- definitions
 - clearing 188
 - displaying 196
 - option of CMS QUERY command 431, 448
- ## FILELIST
- command
 - APPEND option 204
 - example 795
 - FILELIST option 204
 - NOFILELIST option 204
 - PROFILE option 204
 - setting defaults 205
 - option
 - of FILELIST command 204
 - of SENDFILE command 528
- ## FILELIST command 203
- ## filemode
- changing
 - with COPYFILE command 64
 - letter
 - establishing 27
 - replacing 481
 - numbers, changing 490
 - specifying on READCARD command 466
 - specifying when receiving a file 472
- ## FILEPOOL option of CMS SET command 551
- ## FILEPOOL option of the QUERY command 431
- ## files
- alias 73
 - base 73
 - creating
 - with CMS editor 138
 - with COPYFILE command 60
 - with READCARD command 466
 - with the System Product Editor 651
 - defining for CMS/DOS 114
 - display new name for, after renaming 488
 - displaying specified CMS file information via LISTFILE 291
 - identifier
 - entering on FILEDEF command 196
 - entering on LISTDS command 285
 - in command syntax 7
 - inserting lines
 - with UPDATE command 636
 - listing information about 203
 - loading
 - from tape to disk or directory 611
 - from virtual reader to a disk or directory 472
 - modifying 60
 - moving from device to device 345
 - moving with the directory structure 484
 - moving/relocating 484
 - number of CMS files in disk or directory 426
 - overlying data in
 - with COPYFILE command 60
 - packing 62
 - specifying fill character 62

files (continued)

- printing
 - in hexadecimal format 405
 - specifying number of lines per page 404
 - processed by TAPE command, listing 612
 - punched, restoring to disk
 - with DISK LOAD command 108
 - with READCARD command 466
 - with RECEIVE command 472
 - punching to virtual card punch 108
 - reading
 - from virtual card reader. 108
 - receiving from your virtual reader 472
 - relating to OS ddname 186
 - renaming 488
 - replacing lines in
 - with UPDATE command 636
 - replacing old file with new copy 61
 - sorting records in 593
 - tape, writing to disk or directory 611
 - transferring with DISK DUMP command 108
 - unpack 62
 - verifying existence of
 - with ESTATE and ESTATEW commands 600
 - with STATE and STATEW commands 600
 - verifying syntax of file ID
 - with VALIDATE command 649
- ## FILEWAIT option of CMS SET command 553
- ## FILEWAIT option of QUERY command 433
- ## FILL option of COPYFILE command 62
- ## FINIS command 219
- ## fixed-length files, converting to variable-length 67
- ## FLAG option of ASSEMBLE command 37
- ## FLASH option of SETPRT command 590
- ## FMODE
- option of LISTFILE command 294
- ## fn ft fm used to represent file identifier 7
- ## FNAME
- option of LISTFILE command 294
- ## FOR option of COPYFILE command 61
- ## FORMAT command
- command
 - BLKSIZE option 221
 - description 220
 - examples 222
 - LABEL option 221
 - NOERASE option 221
 - performance considerations 222
 - RECOMP option 221
 - selecting appropriate blocksize 222
 - option
 - of LISTDS command 286
 - of LISTFILE command 294
- ## FORMAT MODE (.FO) format word 803
- ## FREE option of LISTDS command 286
- ## FRLABEL option of COPYFILE command 61
- ## FROM option
- of COPYFILE command 61

FROM option (*continued*)
of GENMOD command 227

FSCB Macro
See VM/SP Application Development Reference for CMS

FSCBD Macro
See VM/SP Application Development Reference for CMS

FSCLOSE Macro
See VM/SP Application Development Reference for CMS

FSEQ operand of LABELDEF command 278

FSERASE Macro
See VM/SP Application Development Reference for CMS

FSF tape control function 611

FSOPEN Macro
See VM/SP Application Development Reference for CMS

FSPOINT Macro
See VM/SP Application Development Reference for CMS

FSR tape control function 611

FSREAD Macro
See VM/SP Application Development Reference for CMS

FSSTATE Macro
See VM/SP Application Development Reference for CMS

FSWRITE Macro
See VM/SP Application Development Reference for CMS

FTYPE option of LISTFILE command 294

full-screen CMS
border commands 778
BRKKEY setting, default changed 740
CLEAR key 698, 745
clearing a virtual screen 674
clearing a window 675
connecting a window to a virtual screen 764
copying the physical screen to a CMS file 700
default connections of windows and virtual screens 744
default message routing 744
default virtual screens 743
default windows 741
default windows and virtual screens defined 739
defining a virtual screen 679
defining a window 683
defining fields in a virtual screen 772
deleting a virtual screen definition 686
deleting a window definition 687
dropping a window 688
dropping the WM window 688
entering 739
entering information in a virtual screen 772
erasing data in a virtual screen 674
general information 740

full-screen CMS (*continued*)
location indicator 749
maximizing windows 694
message routing 721
minimizing windows 696
nesting SUSPEND and RESUME 745
null characters, recognizing 555, 737
PA2 key 698, 745
placing a window at top of display order 764
popping a window 697
positioning the cursor in a virtual screen 676
positioning windows 699
querying status of 709
refreshing a screen 719
refreshing a screen from an exec 767
restoring windows 720
resuming full-screen CMS 740
returning to line-mode CMS 739
running applications in full-screen CMS 745
screen display, refreshing from an exec 767
scrolling process 740
sounding the alarm 673
specifying character attributes in a virtual screen 772
suspending temporarily 739
system defaults 741
updating a virtual screen from an exec 767
updating the plane buffers in a virtual screen 772
updating virtual screen with data 770
writing data in a virtual screen 772
writing lines from a file to a virtual screen 690
writing virtual screen data to a CMS file 702

full-screen CMS commands
See windows, CMS commands for

full-screen CMS commands for windowing 669

FULLPROMPT option
of DISK command 109
of READCARD command 466
of RECEIVE command 473

FULLREAD option
of CMS QUERY command 434, 709
of CMS SET command 555, 737

FULLSCREEN option
of CMS QUERY command 709
of CMS SET command 739

G

GEN option
of MACLIB command 331
of TXTLIB command 629

GENDIRT command described 225

GENMOD command
ALL option 228
AMODE option 228
CLEAN option 228
description 226
DOS option 227

GENMOD command (*continued*)
 FROM option 227
 MAP option 227
 NOCLEAN option 228
 NOMAP option 227
 NOSTR option 227
 OS option 227
 RMODE option 229
 STR option 227
 SYSTEM option 228
 TO option 227
GENMSG command
 description 233
 examples 235
 format 233
 messages 235
 operands
 applid 233
 CP 233
 DBCS 233
 fn ft fm 233
 langid 233
 LIST 233
 MARGIN 234
 NODBCS 233
 NOLIST 233
 NOOBJECT 234
 NOXREF 234
 OBJECT 234
 XREF 234
 usage notes 234
GENN operand of LABELF command 278
GENV operand of LABELF command 278
GET sub-function of GLOBALV command 244
GET VSCREEN command 670
 global changes
 querying
 which DOSLIBs were last specified 429
 which MACLIBs were last specified 442
 which TXTLIBs were last specified 450
GLOBAL command
 CSLLIB option 237
 description 237
 DOSLIB option 237
 LOADLIB option 237
 MACLIB option 237
 TXTLIB option 237
 global variables, creating 240
GLOBALV command
 description 240
 examples 246
 GET sub-function 244
 GRPLIST option 244
 GRPSTACK option 244
 INIT option 241
 LIST option 243
 PURGE option 244
 PUT sub-function 243

GLOBALV command (*continued*)
 SELECT option 242
 SESSION file 240
 STACK option 243
 STACKR option 243
 use in CMS execs 246
 GRAF option of FILEDEF command 188
GRANT AUTHORITY command 251
 refid = sfil.by granting authorities 251

H

H border command 780
 handling remote data transmission 579, 753
HB immediate command described 664
 header
 card
 as READ control card 468
 punched by PUNCH command 412
 for LISTFILE command 293
 format 297
HEADER option
 of LISTFILE command 293
 of PUNCH command 412
HELP
 command
 description 255
 format 255
 messages 263
 sample requests 262
 setting defaults 94
 usage notes 260
 format words
 summary 797
 .BX (BOX) 798
 .CM (COMMENT) 800
 .CS (CONDITIONAL SECTION) 801
 .FO (FORMAT MODE) 803
 .IL (INDENT LINE) 805
 .IN (INDENT) 806
 .MT (MENU TYPE) 807
 .OF (OFFSET) 808
 .SP (SPACE LINES) 809
 .TR (TRANSLATE CHARACTER) 810
 Layering options 259
 obtaining additional or related information 343
 operands
 ALL 259
 BRIEF 259
 component-name command-name 256
 DESCRIPT 259
 DETAIL 259
 ERRORS 260
 EXTEND 260
 FORMAT 259
 HELP 256
 menuname MENU 256

HELP (continued)

operands (continued)

MESSAGE message-id 257

MSG 257

NOSCREEN 260

NOTES 260

NOTYPE 260

OPTIONS 260

PARMS 259

RELATED 259

SCREEN 260

taskname TASKS 256

TASKS 255

Other options 260

Subsetting options 259

HELPCONV command

description 265

HEX option

of PRINT command 405

of TYPE command 633

hexadecimal

display in file 633

printing file in 405

HI (Halt Interpretation) immediate command 665

hidden windows, displaying information about 709

HIDE option

of CMS QUERY command 709

HIDE WINDOW command 670

description 692

format 692

messages 693

operands

col 692

line 692

vname 692

wname 692

usage notes 692

hiding a window 692

HISTORY option of EXECUPDT command 180

HNDEXT Macro

See VM/SP Application Development Reference for CMS

HNDINT Macro

See VM/SP Application Development Reference for CMS

HNDSVC Macro

See VM/SP Application Development Reference for CMS

HO immediate command described 665

HT immediate command described 665

HX immediate command

Immediate command 666

effect on DLBL definitions 115

effect on FILEDEF definitions 194

HX immediate command described 666

I

ICS control statement

See include control section (ICS) loader control statement

ID CARD, CP, example 469

ID operand

of TAPEMAC command 619

of TAPPDS command 623

IDENTIFY command

description 267

display user information 267

FIFO option 267

LIFO option 267

STACK option 267

TYPE option 267

IEBTPCH utility program, creating CMS files from tapes created by 622

IEBUPDTE utility program, creating CMS files from tapes created by 622

IEHMOVE utility program

creating CMS files from tapes created by 622

creating CMS MACLIBs from tapes created by 619

IGN option of ASSGN command 44

with DUMMY data sets 118

IJSYSCL defining in CMS/DOS 117

IJSYSCT defined 125

in CMS/DOS 121

IJSYSRL defining in CMS/DOS 117

IJSYSSL defining in CMS/DOS 117

IJSYSUC defined 125

in CMS/DOS 121

IMESCAPE option

of CMS QUERY command 434

of CMS SET command 557

IMMCMD

command

CLEAR operand 269

description 269

QUERY operand 269

SET operand 269

STATUS operand 269

operand of NUCXLOAD command 386

IMMCMD Macro

See VM/SP Application Development Reference for CMS

immediate commands

canceling 269

creating 269

HB 664

HI 665

HO 665

HT 665

HX 666

RO 666

RT 666

SO 667

immediate commands (*continued*)
 summary 12
 TE 667
 TS 667
 IMPCP option
 of CMS QUERY command 434
 of CMS SET command 558
 IMPEX option
 of CMS QUERY command 434
 of CMS SET command 559
 implied
 CP function
 query status of 434
 setting 558
 EXEC function
 query status of 434
 setting 558
 INC option of UPDATE command 637
 INCLUDE command
 AUTO option 273
 CLEAR option 271
 description 271
 DUP option 273
 effect on loader tables 567
 examples 274
 following LOAD command 274
 HIST option 273
 identify TXTLIBs to be searched 237
 INV option 272
 LIBE option 273
 MAP option 272
 NOAUTO option 273
 NOCLEAR option 271
 NODUP option 273
 NOHIST option 273
 NOINV option 272
 NOLIBE option 273
 NOMAP option 272
 NOREP option 273
 NOTYPE option 272
 ORIGIN option 272
 REP option 272
 RESET option 272
 RLDSAVE option 273
 SAME option 273
 START option 273
 TYPE option 272
 include control section (ICS) loader control
 statement 319
 INCR option of XEDIT command 654
 INDENT LINE (.IL) format word 805
 INDENT (.IN) format word 806
 INIT option
 of GLOBALV command 241
 of SETPRT command 590
 INMOVE default for MOVEFILE command
 ddname 345

INPUT
 option
 of CMS QUERY command 435
 of CMS SET command 560
 INSERT control statement for UPDATE
 command 639
 instructions
 altering
 with LOAD command 321
 INSTSEG option
 of CMS QUERY command 435
 of CMS SET command 561
 Interactive Problem Control System (IPCS) 1
 INV option
 of INCLUDE command 272
 of LOAD command 309
 IPCS (Interactive Problem Control System) 1
 issuing a message from a repository 658
 ITEMCT option of TAPEMAC command 620

K

KEY option
 of CMS QUERY command 710
 KEYLEN option of FILEDEF command 191
 KEYPROTECT option
 of CMS SET command 562
 KEYS
 changing CMS storage keys 589
 CMSSTOR RELEASE 562
 DMSFRET 562
 status of settings 435
 keys, user
 protecting 562
 resetting user keys 562
 key, displaying last pressed 710

L

L border command 780
 LABEL option
 of FORMAT command 221
 of LISTFILE command 295
 LABELDEF
 command
 CHANGE option 279
 CLEAR operand 277
 CRDTE operand 278
 description 277
 EXDTE operand 278
 FID operand 277
 FSEQ operand 278
 GENN operand 278
 GENV operand 278
 NOCHANGE option 279
 PERM option 279
 SEC operand 278
 VOLID operand 278
 VOLSEQ operand 278

LABELDEF (*continued*)

- operand of CMS QUERY command 436
- LABOFF** operand of FILEDEF command 198
- LANGGEN** command 21
- LANGLIST** option
 - of CMS QUERY command 436
- LANGMERG** command 22
- LANGUAGE** option
 - of CMS QUERY command 437
 - of CMS SET command 563
- languages, national
 - changing the current language 563
 - controlling language translation tables 586
 - displaying language identifiers 436
 - displaying the current language 437
 - displaying translations and synonyms in effect 449
 - suppressing translations and translation synonyms 586
- language, changing the current 563
- LDRTBLS** option
 - of CMS QUERY command 438
 - of CMS SET command 567
- LEAVE** option
 - of FILEDEF command 193
 - of TAPE command 614
- LET** option of LKED command 302
- LIBE** option
 - of INCLUDE command 273
 - of LKED command 303
 - of LOAD command 310
- LIBMAC** option of ASSEMBLE command 37
- libraries
 - OS, macro libraries
 - See* macro libraries, OS
 - VSE
 - assigning logical units 45
 - obtain information about 135
 - VSE core image
 - defining IJSYSCL 117
 - fetching phases from 183
 - VSE procedure
 - copying procedures from 410
 - displaying directories 135
 - displaying procedures from 410
 - printing procedures from 410
 - punching procedures from 410
 - VSE relocatable
 - assigning SYSRLB 499
 - copying modules from 499
 - defining IJSYSRL 117
 - displaying modules from 499
 - link-editing modules from 130
 - printing modules from 499
 - punching modules from 499
 - VSE source statement
 - assigning SYSSSLB 595
 - copying books 595
 - copying macros from 147
 - defining IJSYSSL 117

libraries (*continued*)

- VSE source statement (*continued*)
 - displaying books 595
 - printing books 595
 - punching books 595
- LIBRARY**
 - loader control statement 318
 - option, of CMS QUERY command 438
- LIFO** (last-in/first-out)
 - operand
 - of CONVERT COMMANDS command 56
 - option
 - of IDENTIFY command 267
 - of NAMEFIND command 350
 - of NUCXMAP command 390
 - of RDR command 454
- LIMCT** option of FILEDEF command 192
- line end character
 - defining 568, 748
 - displaying in CMS 439, 710
 - implications when you redefine 568, 748
 - recognizing 568, 748
- LINECOUN** option
 - of ASSEMBLE command 37
 - of PRINT command 404
- LINEDIT** Macro
 - See* VM/SP Application Development Reference for CMS
- LINEND** option
 - of CMS QUERY command 439, 710
 - of CMS SET command 568, 748
- LINENUM** option of NAMEFIND command 350
- lines
 - mode
 - of CMS editor 139
- lines, changing number in a window 766
- LINK** command, accessing minidisks after 30
- link-editing
 - in CMS/DOS 130
 - modules from VSE relocatable libraries 130
 - TEXT files in storage 308
 - TXTLIB members 631
- linkage editor control statements
 - OS
 - read by TXTLIB command 631
 - required format for TXTLIB command 631
 - VSE supported in CMS/DOS 131
- LIST** option
 - of ASSEMBLE command 37
 - of DEFAULTS command 93
 - of GLOBALV command 243
 - of LKED command 303
 - of LOADLIB command 324
 - of OPTION command 392
- LISTDIR** command 282
 - XEDIT option 282
- LISTDS** command
 - description 285

LISTDS command (*continued*)
 examples 287
 EXTENT option 286
 FORMAT option 286
 FREE option 286
 PDS option 286

LISTFILE command 291
 ALLOC option 294
 APPEND option 293
 ARGS option 293
 BLOCKS option 295
 DATE option 294
 EXEC option 293
 FIFO option 293
 FMODE option 294
 FNAME option 294
 FORMAT option 294
 FTYPE option 294
 HEADER option 293
 LABEL option 295
 LIFO option 293
 NOHEADER option 293
 STACK option 293
 TRACE option 293
 XEDIT option 294

listing execs and editor macros in storage 173

LISTING filetype
 created by access method services 33
 created by ASSEMBLE command 37
 controlling 37
 created by ESERV program 147
 printing 404

listing SFS directories 282

LISTIO command
 A option 300
 ALL option 300
 APPEND option 300
 description 300
 EXEC option 300
 PROG option 300
 STAT option 300
 SYS option 300
 SYSxxx option 300
 UA option 300

LISTX option of OPTION command 392

literal substitution with XMITMSG command 658

LKED command
 ALIGN2 option 303
 AMODE option 304
 description 302
 DISK option 303
 LET option 302
 LIBE option 303
 LIST option 303
 MAP option 303
 NAME option 303
 NCAL option 302
 NE option 303

LKED command (*continued*)
 NOPRINT option 303
 NOTERM option 303
 OL option 303
 OVLY option 303
 PRINT option 303
 REFR option 303
 RENT option 303
 REUS option 303
 RMODE option 304
 SIZE option 303
 TERM option 303
 usage 305
 XCAL option 303
 XREF option 303

LOAD
 AMODE option 310
 command
 AUTO option 309
 CLEAR option 309
 description 308
 DUP option 310
 duplicate CSECTs 316
 effect on loader tables 567
 executing program using 316
 identify TXTLIBs to be searched 237
 INV option 309
 LIBE option 310
 MAP option 309
 NOAUTO option 309
 NOCLEAR option 309
 NODUP option 310
 NOINV option 309
 NOLIBE option 310
 NOMAP option 309
 NOREP option 309
 NORLDSAV option 310
 NOTYPE option 309
 ORIGIN option 312
 REP option 309
 RESET option 309
 RLDSAVE option 310
 START option 310
 TYPE option 309
 used with GENMOD command 230

HIST option 310
 NOHIST option 310
 option
 of DISK command 108
 of TAPE command 611

RMODE option 311

load maps
 creating 309
 with INCLUDE command 272
 with LOAD command 309
 displaying 309
 generated by GENMOD command 227
 invalid card images in 309

- load maps (*continued*)
 - of MODULE files, displaying 342
 - replace card images in 272
- load point, specifying 272, 312
- LOADAREA option of CMS QUERY command 439
- LOADAREA option of CMS SET command 569
- loader
 - CMS 316
 - control statements
 - ENTRY statement 318
 - include control section (ICS) statement 319
 - LIBRARY statement 318
 - loader terminate (LDT) statement 319
 - replace (REP) statement 321
 - set location counter (SLC) statement 320
 - set page boundary (SPB) statement 322
 - search order for unresolved references 316
 - search order, for unresolved references 317
 - tables
 - defining storage for 567
 - displaying number of 438
- loader terminate (LDT) loader control statement 319
- loading a saved segment 519
- loading a virtual 3800 printer via SETPRT command 590
- LOADLIB
 - command
 - COMPRESS option 324
 - COPY option 324
 - description 324
 - DISK option 325
 - EXCLUDE SYSIN control statement 325
 - LIST option 324
 - MODIFY option 325
 - PRINT option 325
 - REPLACE option 325
 - SELECT SYSIN control statement 325
 - TERM option 325
 - option
 - of CMS QUERY command 440
 - of GLOBAL command 237
- LOADLIBs, CMS
 - compressing with LOADLIB command 324
 - copying with LOADLIB command 324
 - creating with LKED command 302
 - executing a load module from 394
 - listing with LOADLIB command 324
- LOADMOD command
 - CMS/DOS considerations 329
 - description 328
 - modules with no header records 329
- location indicator
 - displaying in a window 749
 - querying 710
- location of windows, changing 699
- LOCATION option
 - of CMS QUERY command 710
 - of CMS SET command 749
- locks
 - creating a SHARE lock 79
 - creating an EXCLUSIVE lock 79
 - creating an UPDATE lock 79
- log files
 - controlling updating 750
 - for messages 745
 - for warnings 745
 - querying 711
- LOG option
 - of NOTE command 378
 - of RECEIVE command 473
 - of SENDFILE command 528
- LOGFILE option
 - of CMS QUERY command 711
 - of CMS SET command 750
- logging of messages 745
- logging of warnings 745
- logical
 - line end symbol 568, 748
 - record length of CMS file, defaults used by CMS editor 139
 - units
 - assigning 43
 - ignoring assignments 44
 - listing 300
 - unassigning 545
 - unassigning in CMS/DOS 45
- LONG
 - option of NOTE command 378
- LOWCASE option
 - of ASSGN command 44
 - of COPYFILE command 62
 - of FILEDEF command 193
- lowercase letters
 - translating to uppercase
 - with COPYFILE command 62
 - with PRINT command 404
- LRECL option
 - of COPYFILE command 62
 - of EDIT command 139
 - of FILEDEF command 191

M

- M border command 781
- machine code 158, 166
- MACLIB
 - command
 - ADD option 331
 - COMP option 331
 - DEL option 331
 - description 331
 - DISK option 332
 - FIFO option 332
 - GEN option 331
 - LIFO option 332
 - MAP option 331
 - PRINT option 332

MACLIB (*continued*)
 command (*continued*)
 reading files created by ESERV program 148
 REP option 331
 STACK option 332
 TERM option 332
 XEDIT option 332
 files
 creating 331
 displaying names of MACLIBs to be searched 442
 distributed with CMS system 333
 specifying for assembly or compilation 237
 option
 of CMS QUERY command 442
 of GLOBAL command 237
MACLIST command
 description 335
 setting defaults 94
MACRO
 files created by ESERV program 147
 filetype
 adding to MACLIBs 331
 invalid records handled by MACLIB command 333
 macro definitions
 in assembler listing 37
 in MACRO files 332
 macro libraries
 CMS
 adding to 331
 compact members of 331
 creating 331
 deleting members of 331
 display information about members in 331
 printing members 405
 punching members 412
 reading OS macro libraries into 619
 replacing members of 331
 typing members 633
 creating
 from OS partitioned data sets on tape 619
 from tapes created by IEHMOVE utility program 619
 identifying for assembly 41, 237
 OS
 concatenating 192
 reading into CMS MACLIBs 619
 using in CMS 41
 VSE, copying macros from 147
MAKEBUF command
 description 341
 return code effect on &ERROR statement 341
 managing saved segments 517
MAP
 filetype
 created by DOSLIB command 128
 created by DSERV command 136
 created by LOAD command 315

MAP (*continued*)
 filetype (*continued*)
 created by MACLIB command 332
 created by TAPE command 612
 created by TXTLIB command 630
 option
 of DOSLIB command 128
 of GENMOD command 227
 of INCLUDE command 272
 of LKED command 303
 of LOAD command 309
 of MACLIB command 331
 of TXTLIB command 629
 maps
 created by DOSLIB command 128
 created by GENMOD command 227
 created by LOAD command 309
 created by MACLIB command 331
 created by TXTLIB command 629
 linkage editor in CMS/DOS 131
 master catalog (VSAM)
 identifying in CMS/DOS 121
 identifying (OS VSAM) 125
 master file directory
 contents of 30
 suppressing updating after RENAME command 490
 updating entries in 489
 updating on disk 482
MAXIMIZE WINDOW command 670
 description 694
 format 694
 messages 694
 operands
 wname 694
 = (equal sign) 694
 usage notes 694
 maximizing a window size to the physical screen size 694
MAXTEN option of TAPPDS command 624
MCALL option of ASSEMBLE command 37
MEMBER option
 of FILEDEF command 192
 of PRINT command 404
 of PUNCH command 412
 of TYPE command 633
 of XEDIT command 652
MENU TYPE format word 807
MERGE option of XEDIT command 653
 message logging
 controlling 750
 messages 745
 querying 711
 warnings 745
MESSAGE operand of HELP command 256
 message repository files
 activating 563
 checking for syntax errors 234

- message repository files (*continued*)
 - compiling 233
 - converting into internal form 233
 - example of compiling 235
 - issuing a message 658
 - loading a message repository 234
 - loading user repositories 564
 - output files from conversion 234
 - retrieving a message 658
- message routing
 - default settings 722
 - directing 721
 - querying 712
- minidisks
 - See also* disks
 - counting cylinders on 220
- MINIMIZE WINDOW command 670
 - description 696
 - format 696
 - messages 696
 - operands
 - wname 696
 - = (equal sign) 696
 - usage notes 696
- MINPROMPT option
 - of DISK command 109
 - of READCARD command 466
 - of RECEIVE command 473
- MLOGIC option of ASSEMBLE command 37
- MODESET option of TAPE command 611
- MODIFY option
 - of LOADLIB command 325
 - of SETPRT command 590
- MODMAP command described 342
- MODULE file
 - creating 226
 - debugging 329
 - defining synonyms for 606
 - executing with RUN command 514
 - format 226
 - generating 226
 - loading into storage for execution 328
 - mapping 342
 - VSE, link-editing 130
- modules, VSE, link-editing 130
- MOREHELP command
 - description 343
 - format 343
 - messages 344
 - operands
 - ALL 343
 - BRIEF 343
 - DESCRIPT 343
 - DETAIL 343
 - ERRORS 344
 - FORMAT 343
 - NOTES 344
 - OPTIONS 344
 - PARMS 344

- MOREHELP command (*continued*)
 - operands (*continued*)
 - RELATED 343
 - usage notes 344
- MOVEFILE command
 - default device attributes 347
 - description 345
 - examples 345
 - PDS option 345
- moving a window up in the order of displayed windows 697
- moving directories 484
- moving files 484
- moving windows on a virtual screen 724
- MULT option of DLBL command 116
- multiple
 - extents
 - input files
 - for UPDATE command 637
 - with COPYFILE command 65
 - output files
 - with COPYFILE command 65
 - with RENAME command 489
 - specifying 123
 - specifying in CMS/DOS 119
- multivolume data sets, displaying volumes on which they reside 120
- multivolume VSAM extents
 - identifying with DLBL command 124
 - in CMS/DOS 120
 - maximum number of disks 125
 - in CMS/DOS 120
 - rules for specifying 124
 - in CMS/DOS 120

N

- N border command 781
- NAME option of LKED command 303
- NAMEFIND command
 - description 349
 - FIFO option 350
 - FILE option 350
 - LIFO option 350
 - LINENUM option 350
 - NAMES file format 351
 - NAMES file tags 352
 - sample names file 355
 - SIZE option 350
 - STACK option 349
 - START option 350
 - TYPE option 350
 - XEDIT option 350
- NAMES command
 - description 358
 - nickname 358
 - PF keys on NAMES menu 361
 - sample NAMES screen 362

naming a virtual screen 681
 naming CMS files 4
 naming SFS directories 4
 National Language Support
 See languages, national
 NCAL option of LKED command 302
 NE option of LKED command 303
 NETDATA command 364
 never-call function, specifying in CMS TEXT file 318
 NEWDATE option
 of COPYFILE command 61
 of RECEIVE command 474
 NEWFILE option of COPYFILE command 61
 nickname assigned in NAMES file 358
 NL operand of FILEDEF command 198
 NO option of START command 597
 NOACK option
 of NOTE command 377
 of SENDFILE command 528
 NOALIGN option of ASSEMBLE command 40
 NOALOGIC option of ASSEMBLE command 37
 NOAUTO option
 of INCLUDE command 273
 of LOAD command 309
 NOCC option of PRINT command 404
 NOCHANGE option
 of DLBL command 116
 of FILEDEF command 191
 of LABELDEF command 279
 NOCLEAR option
 of INCLUDE command 271
 of LOAD command 309
 of XEDIT command 652
 NOCOL1 option of TAPPDS command 623
 NOCOMMENTS option of EXECUPDT
 command 181
 NOCOMPRESS option of EXECUPDT command 181
 NOCTL option
 of UPDATE command 637
 of XEDIT command 653
 NODECK option
 of ASSEMBLE command 38
 of OPTION command 392
 NODISK option of ACCESS command 28
 NODUMP option of OPTION command 392
 NODUP option
 of INCLUDE command 273
 of LOAD command 310
 NOEND option of TAPPDS command 624
 NOEOV option of FILEDEF command 193
 NOERASE option of FORMAT command 221
 NOERRS option of OPTION command 393
 NOESD option of ASSEMBLE command 37
 NOFILELIST option of FILELIST command 204
 NOHEADER option
 of LISTFILE command 293
 of PUNCH command 412
 NOHISTORY option of EXECUPDT command 180
 NOINC option of UPDATE command 637
 NOINV option
 of INCLUDE command 272
 of LOAD command 309
 NOLIBE option
 of INCLUDE command 273
 of LOAD command 310
 NOLIBMAC option of ASSEMBLE command 38
 NOLIST option
 of ASSEMBLE command 37
 of OPTION command 392
 NOLISTX option of OPTION command 392
 NOLOG option
 of NOTE command 378
 of RECEIVE command 473
 of SENDFILE command 528
 NOMAP option
 of GENMOD command 227
 of LOAD command 309
 NOMAXTEN option of TAPPDS command 624
 NOMCALL option of ASSEMBLE command 37
 NOMLOGIC option of ASSEMBLE command 37
 NOMSG
 option of XEDIT command 652
 non-relocatable modules in CMS 226
 NONDISP option
 of CMS QUERY command 442, 711
 of CMS SET command 572, 752
 nondisplayable character
 defining character used in place of 572, 752
 displaying character used in place of 442, 711
 NONSHARE option of CMS SET command 573
 nonshared copy
 of named system, obtaining 573
 NONUM option of ASSEMBLE command 39
 NOOBJECT option of ASSEMBLE command 39
 NOPDS option of TAPPDS command 623
 NOPRINT option
 of ASSEMBLE command 38
 of LKED command 303
 of TAPE command 612
 NOPROF option of ACCESS command 28
 NOPROFIL option of XEDIT command 652
 NOPROMPT option
 of DISK command 109
 of READCARD command 466
 of RECEIVE command 473
 NOPROMPT option of COPYFILE command 61
 NORENT option of ASSEMBLE command 40
 NOREP option
 of INCLUDE command 273
 of LOAD command 309
 of UPDATE command 636
 NOREPLACE option
 of DISK option 109
 of READCARD command 467
 of RECEIVE command 473

NORLD option of ASSEMBLE command 37
 NOSAVE option of ACCESS command 28
 NOSCREEN option of XEDIT command 652
 NOSEQ8 option
 of UPDATE command 637
 of XEDIT command 653
 NOSID option of EXECUPDT command 181
 NOSPECS option of COPYFILE command 61
 NOSTD option of SYNONYM command 606
 NOSTK option of UPDATE command 637
 NOSTMT option of ASSEMBLE command 39
 NOSTOR option of UPDATE command 638
 NOSTR option of GENMOD command 227
 NOSYM option of OPTION command 392
 notational conventions 6
 NOTE command
 ACK option 377
 ADD option 377
 CANCEL option 377
 CC option 377
 description 376
 LOG option 378
 LONG option 378
 NOACK option 377
 NOLOG option 378
 NONOTEBOOK option 378
 NOTEBOOK option 377
 PF key settings 380
 PROFILE option 378
 REPLACE option 378
 sending a note 379
 SHORT option 378
 NOTE option of SENDFILE command 528
 NOTEBOOK option
 of NOTE command 377
 of RECEIVE command 473
 NOTERM option
 of ASSEMBLE command 39
 of LKED command 303
 of OPTION command 393
 of UPDATE command 637
 NOTEST option of ASSEMBLE command 39
 NOTRC option of PRINT command 404
 NOTRUNC option COPYFILE command 62
 NOTYPE option
 of COPYFILE command 61
 of ERASE command 142
 of INCLUDE command 272
 of LOAD command 309
 of RDR command 454
 of RENAME command 489
 of SENDFILE command 529
 NOUPDATE option
 of EXECUPDT command 181
 of XEDIT command 653
 NOUPDIRT option of RENAME command 489
 NOWTM option of TAPE command 612
 NOXREF option
 of ASSEMBLE command 38
 of OPTION command 393
 NOYFLAG option of ASSEMBLE command 40
 NSL operand
 of FILEDEF command 198
 of TAPEMAC command 619
 of TAPPDS command 623
 NUCEXT Function
 See VM/SP Application Development Reference for CMS
 nucleus
 CMS protected storage 575
 protection feature
 displaying status of 443
 setting 575
 nucleus extensions
 cancel an extension 384
 installation 389
 obtain information about 389
 NUCXDROP command
 description 384
 NUCXLOAD command
 description 385
 ENDCMD operand 386
 IMMCMD operand 386
 PUSH operand 386
 SERVICE operand 386
 SYSTEM operand 385
 NUCXMAP command
 description 389
 FIFO option 390
 LIFO option 390
 STACK option 390
 null
 line
 when entering VSAM extents 124
 when entering VSAM extents in CMS/DOS 119
 null characters, recognizing in full-screen CMS 555,
 737
 NUMBER option of ASSEMBLE command 39
 numeric
 substitution with XMITMSG command 658

O
 O border command 781
 object deck, assembler, generating 38
 OBJECT option of ASSEMBLE command 38
 obtaining additional or related help 343
 OFFSET (.OF) format word 808
 OL option of LKED command 303
 OLD option of SENDIFLE command 532
 OLDDATE option
 of COPYFILE command 61
 of DISK LOAD command 109
 of RECEIVE command 474

- operands, command 2
- Operating System (OS)
 - data sets
 - defining in CMS 186
 - listing information 285
 - disks, accessing 31
 - environment, resetting 176
 - linkage editor control cards, adding to TEXT files 631
 - macro libraries
 - reading into CMS MACLIBs 619
 - used in assembly 41
 - option of GENMOD command 227
 - partitioned data sets
 - See partitioned data sets (PDS)
 - tapes
 - containing partitioned data sets 623
 - standard-label processing 625
 - utility programs
 - creating CMS files from tapes created by 622
 - IEBTPCH 622
 - IEBUPDTE 622
 - IEHMOVE 622
- OPTCD option of FILEDEF command 192
- OPTION
 - command
 - DECK option 392
 - description 392
 - DUMP option 392
 - ERRS option 393
 - LIST option 392
 - LISTX option 392
 - NODECK option 392
 - NODUMP option 392
 - NOERRS option 393
 - NOLIST option 392
 - NOLISTX option 392
 - NOSYM option 392
 - NOTERM option 393
 - NOXREF option 393
 - SYM option 392
 - TERM option 393
 - XREF option 393
 - 48C option 393
 - 60C option 393
 - option, of CMS QUERY command 443
- options
 - command 2
 - for DOS/VS COBOL compiler in CMS/DOS, querying 443
 - for DOS/VS COBOL compiler, specifying 392
 - LOAD and INCLUDE commands, retaining 273
- ORIGIN
 - option
 - of FETCH command 183
 - of INCLUDE command 272
 - of LOAD command 312

- OS (Operating System)
 - data sets
 - defining in CMS 186
 - listing information 285
 - disks, accessing 31
 - environment, resetting 176
 - linkage editor control cards, adding to TEXT files 631
 - macro libraries
 - reading into CMS MACLIBs 619
 - used in assembly 41
 - option of GENMOD command 227
 - partitioned data sets
 - See partitioned data sets (PDS)
 - tapes
 - containing partitioned data sets 623
 - standard-label processing 625
 - utility programs
 - creating CMS files from tapes created by 622
 - IEBTPCH 622
 - IEBUPDTE 622
 - IEHMOVE 622
- OSRUN
 - description 394
 - PARM keyword 394
- OUTMODE option of UPDATE command 637
- OUTMOVE default for MOVEFILE command
 - ddname 345
- OUTPUT option
 - of CMS SET command 574
 - option
 - of CMS QUERY command 443
- OVLY option
 - of COPYFILE command 61
 - of LKED command 303

P

- P border command 782
- PACK
 - option of COPYFILE command 62
- parameter list
 - passed by RUN command 515
 - passed by START command 598
 - passed to SVC instruction, recorded 602
- parent minidisk of read-only extension 27
- parentheses
 - before option list 3
- PARM
 - keyword of OSRUN command 394
- PARSECMD command
 - description 395
 - format 395
 - messages 397
 - operands
 - APPLID 395
 - NOTYPE 395
 - STRING 395
 - TYPE 395

PARSECMD command (*continued*)
 operands (*continued*)
 uniqueid 395
 usage notes 396
 parsing facility
 See also DLCS (Definition Language for Command Syntax)
 calling from an exec 395
 variables returned to EXEC 396
 partitioned data sets (PDS)
 copying files into CMS files 345
 copying into partitioned data sets 345
 displaying member names 286
 listing members of 285
 on tapes, creating CMS files 623
 PA1 key in full-screen CMS 745
 PA2 key in full-screen CMS 698, 745
 PD option of DSERV command 135
 PDS option
 of LISTDS command 286
 of MOVEFILE command 345
 of TAPPDS command 623
 PDS (partitioned data sets)
 copying files into CMS files 345
 copying into partitioned data sets 345
 displaying member names 286
 listing members of 285
 on tapes, creating CMS files 623
 PEEK command
 description 398
 PF key settings 399
 PROFILE option 398
 PERM option
 of DLBL command 115
 of FILEDEF command 191
 of LABELDEF command 279
 permanent file definitions 191
 PF key default settings
 changing for full-screen CMS 734
 changing for the WM window 761
 displaying settings for full-screen CMS 707
 displaying settings for the WM window 716
 for full-screen CMS 734
 for the WM window 761
 on NAMES menu 361
 on NOTE menu 380
 on PEEK screen 399
 on RDRLIST screen 462
 on SENDFILE menu 530
 PF keys
 changing settings for full-screen CMS 734
 changing settings for WM window 761
 displaying definitions for full-screen CMS 707
 displaying definitions for WM window 716
 phase library
 clearing to zeros 132
 CMD/DOS 128
 deleting phases from 128
 phases
 executing in CMS/DOS 183
 in VSE core image libraries, obtaining information about 135
POP WINDOW command 670
 description 697
 format 697
 messages 698
 operands
 n 697
 WM 697
 wname 697
 * (asterisk) 697
 usage notes 697
 popping a variable size window 697
 popping a window 697
 popping the WM window 697
POSITION WINDOW command 670
 description 699
 format 699
 messages 699
 operands
 pscol 699
 psline 699
 wname 699
 usage notes 699
 positioning the cursor in a virtual screen 676
 positioning windows 699
 preferred auxiliary files 645
 prefixes
 identifying sets of files
 with ACCESS command 30
PRINT
 command
 CC option 404
 description 403
 HEX option 405
 LINECOUN option 404
 MEMBER option 404
 NOCC option 404
 NOTRC option 404
 OVERSIZE option 403
 TRC option 404
 UPCASE option 404
 option
 of AMSERV command 33
 of ASSEMBLE command 38
 of DOSLIB command 128
 of DOSLKED command 131
 of DSERV command 136
 of EXECIO command 158
 of LKED command 303
 of LOADLIB command 325
 of MACLIB command 332
 of PSERV command 410
 of RSERV command 499
 of SSERV command 595
 of TAPE command 612
 of TXTLIB command 630

PRINT (*continued*)
 option (*continued*)
 of UPDATE command 637
PRINTER option
 of ASSGN command 43
 of FILEDEF command 187
PRINTL Macro
 See VM/SP Application Development Reference for CMS
 private libraries
 See libraries, VSE
PROC files creating in CMS/DOS 410
 procedures, VSE, copying into CMS files 410
PROFILE EXEC, suppressing execution of 28
PROFILE option
 of NOTE command 378
 of PEEK command 398
 of RDRLIST command 459
 of XEDIT command 652
PROG option of LISTIO command 300
PROGMAP command 407
 program function (PF) key
 See PF key default settings
 program handling commands
 PROGMAP 407
 RTNLOAD 503
 RTNMAP 507
 RTNSTATE 511
 program stack
 buffer
 creating 341
 eliminating 134
 using WAITRD function to read lines from 341
 determining number of lines in 536
 programmer logical units
 for job catalogs 117
 listing assignments for in CMS/DOS 300
 restrictions for R and T 45
 valid assignments in CMS/DOS 45
 programs
 compilation and execution, with RUN command 514
 entry point
 selection during CMS loader processing 316
 specifying 308
 execution
 halting 666
 in CMS/DOS 183
 with INCLUDE command 271
 with LOAD command 308
 with START command 597
 loading into storage
 with INCLUDE command 271
 stack buffer, clearing 100
 programs loaded in storage
 displaying information 407
PROMPT
 option of COPYFILE command 61

PROTECT option
 of CMS QUERY command 443
 of CMS SET command 575
PSERV command
 description 410
 DISK option 410
 PRINT option 410
 PUNCH option 410
 TERM option 410
PUNCH
 assembler punch output ddname 41
 command
 description 412
 HEADER card format 413
 HEADER option 412
 MEMBER option 412
 NOHEADER option 412
 option
 of ASSIGN command 43
 of EXECIO command 157
 of FILEDEF command 188
 of PSERV command 410
 of RSERV command 499
 of SSERV command 595
PUNCHC Macro
 See VM/SP Application Development Reference for CMS
 punched files, restoring to disk or directory 108
PURGE option
 of GLOBALV command 244
 of RECEIVE command 473
 purging a saved segment 522
PUSH option of NUCXLOAD command 386
PUT SCREEN command 670
 description 700
 format 700
 messages 700
 option
 fn ft fm 700
 usage notes 700
PUT sub-function of GLOBALV command 243
PUT VSCREEN command 670
 description 702
 format 702
 messages 702
 option
 fn ft fm 702
 fromlin 702
 numlin 702
 vname 702
 usage notes 702

Q
QUERY command (CMS)
 ABBREV option 417
 ACCESSED option 418
 APL option 421, 705

QUERY command (CMS) (continued)

AUTOREAD option 424
BLIP option 424
BORDER option 706
CHARMODE option 706
CMSLEVEL option 424
CMSPF option 707
CMSTYPE option 424
CURSOR option 707
description 415, 704
DISK option 425
DISPLAY option 708
DLBL option 427
DOS option 428
DOSLIB option 429
DOSLNCNT option 429
DOSPART option 429
EXECTRAC option 430
FIFO option 451, 717
FILEDEF option 431, 448
FULLREAD option 434, 709
FULLSCREEN option 709
HIDE option 709
IMESCAPE option 434
IMPCP option 434
IMPEX option 434
INPUT option 435
INSTSEG option 435
KEY option 710
KEYPROTECT option 435
LABELDEF option 436
LANGLIST option 436
LANGUAGE option 437
LDRTBLS option 438
LIBRARY option 438
LIFO option 451, 717
LINEND option 439, 710
LOADAREA option 439
LOADLIB option 440
LOCATION option 710
LOGFILE option 711
MACLIB option 442
NONDISP option 442, 711
OPTION option 443
OUTPUT option 443
PROTECT option 443
RDYMSG option 443
REDTYPE option 443
RELPAGE option 444
REMOTE option 444, 711
RESERVED option 712
ROUTE option 712
SEARCH option 444
SEGMENT option 445
SHOW option 713
STACK option 451, 717
STROECLR option 447
SYNONYM option 448

QUERY command (CMS) (continued)

SYSNAMES option 448
TEXT option 449, 713
TRANSLATE option 449
TXTLIB option 450
UPSI option 451
VSCREEN option 714
WINDOW option 715
WMPF option 716
QUERY LOADAREA command 439
QUERY operand of IMMCMD command 269
QUERY STORECLR command 447
querying remote data transmission 444, 711
querying status of full-screen CMS environment 704
querying status of virtual machine environment 415

R

R border command 782
RD option of DSERV command 135
RDCARD Macro
See VM/SP Application Development Reference for CMS
RDR command
description 454
FIFO option 454
LIFO option 454
NOTYPE option 454
STACK option 454
use of = 454
RDRLIST command
APPEND option 459
default PF key settings 462
description 459
displaying a file 463
issuing commands from RDRLIST 461
PROFILE option 459
special symbols 461
synonyms that sort the list 462
RDTAPE Macro
See VM/SP Application Development Reference for CMS
RDTERM Macro
See VM/SP Application Development Reference for CMS
RDYMSG option
of CMS QUERY command 443
of CMS SET command 576
READ control card
deleting 468
format of 468
read-only
extensions
editing files on 138
releasing 482
READCARD command
description 466
FULLPROMPT option 466

READCARD command (*continued*)
 MINPROMPT option 466
 NOPROMPT option 466
 NOREPLACE option 467
 REPLACE option 466
 READER option
 of ASSGN command 43
 of FILEDEF command 188
 reader, virtual
 determine characteristics of next file in 454
 information about files in 459
 listing the files in 459
 PEEK at a file in 398
 reading a file from 472
 receiving a file from 472
 reading a real card deck 468
 ready message
 format 576
 long form 576
 querying setting of 443
 setting 576
 short form 576
 special format in exec 150
 read/write
 status of disks
 controlling 27
 listing for disk assignments in CMS/DOS 300
 querying 425
 read, console, after a CMS command 540
 real card deck, reading into virtual card reader 468
 RECEIVE command
 acknowledge receipt of file 475
 description 472
 FULLPROMPT option 473
 LOG option 473
 MINPROMPT option 473
 NEWDATE option 474
 NOLOG option 473
 NOPROMPT option 473
 NOREPLACE option 473
 NOTEBOOK fn option 473
 NOTEBOOK * option 473
 OLDDATE option 474
 PURGE option 473
 REPLACE option 473
 STACK option 474
 RECFM
 option
 of COPYFILE command 62
 of FILEDEF command 191
 RECOMP option of FORMAT command 221
 record format
 of CMS file
 changing 62, 67
 listing 294
 of file, specifying 191
 records that can be punched 413
 record length
 default used by CMS editor 139
 modifying 139
 of CMS file
 changing 62
 listing 294
 maximum lengths for PRINT command 405
 specifying with FILEDEF command 191
 records
 displaying selected positions of 633
 in file, numbering with UPDATE command 636
 red type
 for error messages 577
 REDTYPE option
 of CMS QUERY command 443
 of CMS SET command 577
 reducing a window size to one line 696
 references
 unresolved
 resolving with INCLUDE command 273
 resolving with LOAD command 316
 REFR option of LKED command 303
 REFRESH command 671
 description 719
 format 719
 messages 719
 refreshing a screen from an exec 767
 refreshing a screen in full-screen CMS 719
 REGEQU Macro
 See VM/SP Application Development Reference for CMS
 RELATED HELP
 described 258
 obtaining using HELP command 255
 obtaining using MOREHELP command 343
 RELEASE command
 description 481
 DET operand 481
 DET option 481
 freeing an accessed disk or directory 481
 restriction of read-only extensions 482
 releasing segment space 524
 relocatable
 libraries (VSE), displaying directories of 135
 modules, link-editing in CMS/DOS 130
 RELOCATE command 484
 relocating directories 484
 relocating files 484
 relocation dictionary assembler 37
 RELPAG option
 of CMS QUERY command 444
 of CMS SET command 578
 remote data transmission, handling 579, 753
 remote data transmission, querying 444, 711
 REMOTE option
 of CMS QUERY command 444, 711
 of CMS SET command 579, 753

- removing a virtual screen definition 686
- removing a window definition 687
- removing execs and editor macros from storage 153
- RENAME command 488
 - NOTYPE option 489
 - NOUPDIRT option 489
 - TYPE option 489
 - UPDIRT option 489
- RENT
 - option
 - of ASSEMBLE command 40
 - of LKED command 303
- REP option
 - of INCLUDE command 272
 - of LOAD command 309
 - of MACLIB command 331
 - of UPDATE command 636
- REPLACE
 - control statement, for UPDATE command 640
 - of READCARD command 466
 - option
 - of COPYFILE command 61
 - of DISK command 109
 - of LOADLIB command 325
 - of NOTE command 378
 - of RECEIVE command 473
- replace (REP)
 - image of statement in load map 272
 - loader control statement 321
- RESERVE command
 - description 493
 - format of RESERVED file 493
 - of CMS QUERY command 712
 - use with DISKID function 493
- reserved lines
 - defined 754
 - displaying number in a window 712
 - specifying number in a window 754
- RESERVED option
 - of CMS SET command 754
- reserving segment space 525
- RESET
 - option
 - of INCLUDE command 272
 - of LOAD command 309
- resetting
 - OS environment 176
 - VSAM environment 176
- RESTORE WINDOW command 671
 - description 720
 - format 720
 - messages 720
 - operands
 - wname 720
 - = (equal sign) 720
- restoring a window 720
- restrictions
 - access method services and VSAM
 - OS/VS users 817
- restrictions (*continued*)
 - access method services and VSAM (*continued*)
 - VSE users 817
- resuming full-screen CMS 740
- retrieving a message from a repository 658
- return code
 - from MAKEBUF command effect on &ERROR statement 341
 - from SENTRIES effect on exec procedure 536
- return codes
 - CMS in exec procedure 150
 - from access method services 35
 - from CMS exec interpreter 150
 - from EXEC 2 interpreter 151
 - from System Product Interpreter 151
- returning a segment space to CMS 524
- returning to full-screen CMS after suspending 740
- REUS option of LKED command 303
- REVOKE AUTHORITY command 496
- revoked alias
 - definition of 74
- REW tape control function 611
- REWIND option
 - of TAPE command 614
- RLD option of ASSEMBLE command 37
- RLDSAVE option
 - of INCLUDE command 273
 - of LOAD command 310
- RO immediate command 666
- ROUTE command 671
 - description 721
 - format 721
 - messages 723
 - of CMS QUERY command 712
 - operands
 - ALARM 721
 - CMS 721
 - CP 721
 - MESSAGE 721
 - msgclass 721
 - NETWORK 721
 - NOALARM 721
 - NONOTIFY 722
 - NOTIFY 721
 - SCIF 721
 - vname 721
 - WARNING 721
 - * (asterisk) 721
 - usage notes 722
- routines, CSL
 - binding 503
 - displaying 507
 - loading 503
 - See also the VM/SP Application Development Reference for CMS
 - verifying the existence of CSL routines 511
- routing messages 721

RSERV command
 description 499
 DISK option 499
 PRINT option 499
 PUNCH option 499
 TERM option 499
 RT immediate command 666
 RTNDROP command 501
 RTNLOAD command 503
 RTNMAP command 507
 RTNSTATE command 511
RUN
 description 514
 tape control function 611
 running applications in full-screen CMS 745

S

S border command 782
 SAME option of INCLUDE command 273
 saved segments
 assigning logical segments in physical segments 518
 creating a segment space 525
 loading 519
 managing 517
 purging 522
 releasing reserved segment space 524
 reserving segment space 525
 saved segment, CMS commands
 SEGMENT 517
 SEGMENT ASSIGN 518
 SEGMENT LOAD 519
 SEGMENT PURGE 522
 SEGMENT RELEASE 524
 SEGMENT RESERVE 525
 saved segment, saved system names 582
 saved systems
 names
 querying 448
 setting 582
 sharing 582
 SAVEONLY option of ACCESS command 28
 SCAN option of TAPE command 611
 screen display, refreshing from an exec 767
 screen images, copying to a CMS file 700
 screen, copying the image to a CMS file 700
 screen, displaying characteristics of 708
SCROLL command 671
 description 724
 format 724
 messages 726
 operands
 BACKWARD 724
 BOTTOM 725
 DOWN 725
 FORWARD 725
 LEFT 725
 NEXT 725
 RIGHT 725

SCROLL command (*continued*)
 operands (*continued*)
 TOP 725
 UP 725
 wname 724
 usage notes 725
 scrolling a window 698, 724
 scrolling process in full-screen CMS 740
 SD option of DSERV command 135
 SEARCH option of CMS QUERY command 444
 search order
 for CMS commands 9
 for CMS loader 316
 for executable phases in CMS/DOS 183
 for relocatable modules in CMS/DOS 131
 of CMS disks, querying 444
 searching
 for CSL routines 503
 SEC operand of LABELDEF command 278
SEGMENT command 517
SEGMENT option
 of CMS QUERY command 445
 SELECT option of GLOBALV command 242
 SELECT SYSIN control statement 325
SENDFILE command
 ACK option 528
 default PF key settings on SENDFILE menu 530
 description 527
 example 533
 file format 531
 FILELIST option 528
 LOG option 528
 NEW option 528
 NOACK option 528
 NOFILELIST option 528
 NOLOG option 528
 NOTE option 528
 NOTYPE option 529
 OLD option 528
 TYPE option 529
 sending
 messages 627
 notes 379
SENTRIES command
 description 536
 effect of non-zero return code on EXECs 536
SEQUENCE control statement for UPDATE
 command 638
 sequence numbers
 assigned to VSAM extents 124
 assigned (CMS/DOS) to VSAM extents 119
SEQ8 option
 of UPDATE command 636
 of XEDIT command 653
SERVER option of the CMS SET command 580
SERVICE operand of NUCXLOAD command 386
SESSION file of GLOBALV command 240

SET command

- description 537, 727
- determining status of SET operands 415, 704
- list of options 537, 727
- messages 537, 727
- operands
 - ABBREV 538
 - APL 539, 728
 - AUTOREAD 540
 - BLIP 541
 - BORDER 729
 - CHARMODE 732
 - CMSPF 734
 - CMSTYPE 542
 - COMDIR 543
 - DOS 545
 - DOSLNCNT 547
 - DOSPART 548
 - EXECTRAC 550
 - FILEPOOL 551
 - FILEWAIT 553
 - FULLREAD 555, 737
 - FULLSCREEN 739
 - IMESCAPE 557
 - IMPCP 558
 - IMPEX 559
 - INPUT 560
 - INSTSEG 561
 - KEYPROTECT 562
 - LANGUAGE 563
 - LDRTBLS 567
 - LINEND 568, 748
 - LOADAREA 569
 - LOCATION 749
 - LOGFILE 750
 - NONDISP 572, 752
 - OUTPUT 574
 - PROTECT 575
 - RDYMSG 576
 - REDTYPE 577
 - RELPAGE 578
 - REMOTE 579, 753
 - RESERVED 754
 - SERVER 580
 - STORECLR 581
 - TEXT 583, 756
 - THRESHOLD 584
 - TRANSLATE 586
 - UPSI 588
 - VSCREEN 757
 - WINDOW 759
 - WMPF 761
- SYSNAME option 582
- usage notes 537, 727
- set location counter (SLC) loader control statement 320
- SET operand
 - of DEFAULTS command 93

SET operand (*continued*)

- of IMMCMD command 269
- set page boundary (SPB) loader control statement 322
- SETPRT command
 - by accessing directories 27
 - by creating aliases 73
 - CHARS option 590
 - COPIES option 590
 - COPYnr option 590
 - description 590
 - FCB option 590
 - FLASH option 590
 - INIT option 590
 - MODIFY option 590
 - Sharing files
 - using 591
 - setting another language 563
 - setting partition size for CMS/DOS 548
- SHARED
 - option
 - of EXECDROP command 153
 - of EXECMAP command 174
- Shared File System (SFS) tasks
 - accessing a directory 27
 - copying files between minidisks and directories 60
 - creating a directory 77
 - creating a lock 79
 - creating a namedef 83
 - creating an alias 73
 - deleting a lock 96
 - deleting a namedef 99
 - displaying list of directories 101
 - displaying search order 444
 - displaying status of accessed directories 418
 - displaying status of file requests 433
 - entering commands from DIRLIST 101
 - erasing a directory 141
 - granting authorities 251
 - issuing XEDIT subcommands from DIRLIST 101
 - listing CMS files 291
 - listing directories 101, 282
 - listing shared files and directories 203
 - naming directories 4
 - relocating CMS files 484
 - renaming files or directories 488
 - requesting a warning message for file space 584
 - revoking authority 496
 - setting a default file pool 551
 - setting a request for wait 553
 - verifying existence of CMS files 600
 - xediting CMS files 651
- SHORT
 - option of NOTE command 378
- SHOW option
 - of CMS QUERY command 713
- SHOW WINDOW command 671
 - description 764
 - format 764

SHOW WINDOW command (*continued*)

- messages 764
- operands
 - col 764
 - line 764
 - vname 764
 - wname 764
- usage notes 764
- SID option of EXECUPDT command 181
- SIDCODE option of XEDIT command 654
- SINGLE option of COPYFILE command 63
- SIZE option of NAMEFIND command 350
- SIZE WINDOW command 671
 - description 766
 - format 766
 - messages 766
 - operands
 - cols 766
 - lines 766
 - wname 766
 - usage notes 766
- SKIP option
 - of TAPE command 611
- SL operand
 - of FILEDEF command 198
 - of TAPEMAC command 619
 - of TAPPDS command 622
- SLC (set location counter) loader control statement 320
- SO immediate command 667
- SORT
 - command
 - description 593
 - storage requirements 593
 - option of DSERV command 136
- sort fields defined 593
- sounding the alarm 673
- source files
 - assembling
 - identifying macro libraries 237
 - for assembler 36
 - updating with EXECUPDT command 180
 - updating with UPDATE command 636
- source file, numbering records with UPDATE command 636
- source statement libraries, VSE, displaying directories 135
- source symbol table, assembler, generating 38
- SPACE LINES (.SP) format word 809
- space, determine free extents for VSAM 285
- special variables
 - See EXEC, special variables
- specification list for COPYFILE command format 69
- SPECS option of COPYFILE command 61
- SPOOL command
 - used with DISK DUMP command 109
 - used with PRINT command 406

SSERV command

- description 595
- DISK option 595
- PRINT option 595
- PUNCH option 595
- TERM option 595

STACK

- option
 - of CONVERT COMMANDS command 56
 - of EXECMAP command 174
 - of GLOBALV command 243
 - of IDENTIFY command 267
 - of NAMEFIND command 349
 - of NUCXMAP command 390
 - of RDR command 454
 - of RECEIVE command 474

STACKR option of GLOBALV command 243

START

- command
 - description 597
 - NO option 597
 - passing arguments 597
- option
 - of FETCH command 183
 - of INCLUDE command 273
 - of LOAD command 310
 - of NAMEFIND command 350

- starting point for execution of module, setting 308

STAT option of LISTIO command 300

STATE command described 600

STATEW command described 600

- status of full-screen CMS environment, querying 704
- status of virtual machine environment, querying 415

STATUS operand of IMMCMD command 269

STD option of SYNONYM command 606

STEM option of EXECIO command 160

STK option of UPDATE command 637

STMT option of ASSEMBLE command 39

STOR option of UPDATE command 637

storage

- clearing to zeros
 - in CMS/DOS 132
 - with LOAD command 309
- freeing storage 482
- initializing for module file execution 226
- releasing pages of, after command execution 578
- requirements for SORT command 593
- specifying for CMS/DOS partition 548
- with INCLUDE command 271

storage-resident EXECs and editor macros

- controlling system searching of CMS installation
 - saved segment 561
- discontinue use of the CMS installation saved segment 153
- listing 173
- querying status of CMS installation saved segment 435
- removing 153

STORECLR option of CMS QUERY command 447
 STORECLR option of the CMS SET command 581
 STR option of GENMOD command 227
 subdirectory
 creating 77
 sublibraries of VSE source statement, copying
 books 595
 substitution
 with XMITMSG command 658
 SUL operand of FILEDEF command 198
 summary
 of HELP and HELPCONV format words 797
 suppressing virtual screen updates 758
 suspending full-screen CMS 739
 SVC
 instructions
 tracing 602
 SVCTRACE command
 description 602
 output 603
 SYM option of OPTION command 392
 symbol
 variable
 See variable symbols
 SYNONYM
 command
 CLEAR option 606
 description 606
 example 608
 NOSTD option 606
 relationship to SET ABBREV command 606
 STD option 606
 option, of CMS QUERY command 448
 synonym table
 clearing 606
 defining 607
 displaying system synonym tables 449
 displaying user synonym tables 449
 format for entries in 607
 invoking 606
 setting system translation synonyms 586
 setting user translation synonyms 586
 synonyms
 for CMS user-written commands
 defining 606
 displaying 609
 example 607
 SYS option of LISTIO command 300
 SYSCAT assigned in CMS/DOS 121
 SYSIN
 assembler input 41
 logical unit assignment in CMS/DOS 43
 SYSIPT assigned for ESERV program 147
 SYSLOG assigned in CMS/DOS 43
 SYSLSLST lines per page
 displaying number of 429
 setting number of 547

SYSNAME option of CMS SET command 582
 SYSNAMES option of CMS QUERY command 448
 SYSPARM option of ASSEMBLE command 40
 SYSRES assigned in CMS/DOS 45
 SYSTEM
 operand of NUCXLOAD command 385
 option
 of CONVERT COMMANDS command 55
 of EXECDROP command 153
 of EXECLOAD command 171
 of EXECMAP command 173
 of GENMOD command 228
 system and programmer logical units entered on DLBL
 command 117
 system disk
 files available 28
 releasing 482
 system logical units
 invalid assignments in CMS/DOS 45
 listing assignments for in CMS/DOS 300
 valid assignments in CMS/DOS 43
 System Product interpreter
 error codes 151
 tracing programs interpreted by 550
 system residence volume, VSE, specifying 545
 SYSTEMM option of ASSEMBLE command 39
 SYSxxx option
 of ASSGN command 43
 of DLBL command 115
 of LISTIO command 300

T

Table Character Reference byte 404
 TAPE command
 BLKSIZE option 612
 control functions
 BSF 611
 BSR 611
 ERG 611
 FSF 611
 FSR 611
 REW 611
 RUN 611
 WTM 611
 DEN option 613
 description 610
 DISK option 612
 DUMP option 610
 DVOL1 option 612
 EOF option 613
 EOT option 613
 LEAVE option 614
 LOAD option 611
 MODESET option 611
 NOPRINT option 612
 NOWTM option 612
 PRINT option 612

- TAPE command (*continued*)
 - REWIND option 614
 - SCAN option 611
 - SKIP option 611
 - TAPn option 613
 - TERM option 612
 - TRANSFER BUFF option 614
 - TRANSFER IMMEDIATE option 614
 - TRTCH option 614
 - WTM option 612
 - WVOL1 option 612
 - 18TRACK option 613
 - 7TRACK option 613
 - 9TRACK option 613
- TAPECTL Macro
 - See VM/SP Application Development Reference for CMS*
- TAPEMAC command
 - description 619
 - ID operand 619
 - ITEMCT option 620
 - NSL operand 619
 - SL operand 619
 - TAPn option 620
- tapes
 - assigning to logical units, in CMS/DOS 43
 - backward spacing 611
 - control functions
 - restrictions when using 615
 - creating CMS disk files 622
 - density of, specifying 613
 - displaying file names on 611
 - dumping and loading files 610
 - erasing a defective section 611
 - files
 - created by OS utility programs 622
 - created by tape command 610
 - writing to a disk or directory 610
 - forward spacing 611
 - labels
 - displaying definitions in effect 436
 - displaying VOL1 label 612
 - in FILEDEF command 188
 - in TAPEMAC command processing 620
 - in TAPPDS command processing 623
 - specifying descriptive information 277
 - writing VOL1 label 612
 - marks
 - writing 612
 - OS, standard label processing 622
 - positioning
 - after VOL1 label is processed 612
 - at specified file 611
 - recording technique, specifying 614
 - rewinding 611
 - used for AMSERV input and output 33
 - entering ddnames 34
 - in CMS/DOS 34
- TAPESL Macro
 - See VM/SP Application Development Reference for CMS*
- TAPIN option of AMSERV command 33
- TAPn option
 - of ASSGN command 44
 - of FILEDEF command
 - description 188
 - usage 198
 - of TAPE command 613
 - of TAPEMAC command 620
 - of TAPPDS command 623
- TAPOUT option of AMSERV command 33
- TAPPDS command
 - COL1 option 623
 - description 622
 - END option 624
 - ID operand 623
 - MAXTEN option 624
 - NOCOL1 option 623
 - NOEND option 624
 - NOMAXTEN option 624
 - NOPDS option 623
 - NSL operand 623
 - PDS option 623
 - processing OS standard-label tapes 624
 - SL operand 622
 - TAPn option 623
 - UPDATE option 623
- TASKS operand of HELP command 255
- TD option of DSERV command 135
- TE (Trace End) immediate command 667
- TELL command
 - change the CP command that TELL uses 627
 - description 627
 - restrictions 627
 - using nicknames 627
- TERM option
 - of DOSLIB command 128
 - of DOSLKED command 131
 - of DSERV command 136
 - of LKED command 303
 - of LOADLIB command 325
 - of MACLIB command 332
 - of OPTION command 393
 - of PSERV command 410
 - of RSERV command 499
 - of SSERV command 595
 - of TAPE command 612
 - of TXTLIB command 630
 - of UPDATE command 637
- TERMINAL option
 - of ASSEMBLE command 39
 - of ASSGN command 44
 - of FILEDEF command 187
- terminals
 - output
 - halting 665
 - restoring 666

terminate execution of System Product interpreter or
 EXEC 2 execs 665
 TEST option of ASSEMBLE command 39
 TEXT
 assembler output ddname 41
 files
 automatic loading 309
 cards read in by loader 318
 creating with assembler 36
 executing with RUN command 514
 link-editing in CMS/DOS 130
 linking in storage 308
 loading into virtual storage 308
 resolving unresolved references with LOAD
 command 316
 libraries
 See TXTLIB
 of CMS QUERY command 449, 713
 of CMS SET command 583, 756
 TEXT character conversion
 activating in CMS 583, 756
 displaying status in CMS 449, 713
 TEXT files
 loading into storage for execution 271
 setting starting point for execution 308
 THRESHOLD option of the CMS SET command 584
 TO
 keyword of SENDFILE command 527
 option of GENMOD command 227
 TODACCNT Function
 See VM/SP Application Development Reference for
 CMS
 TOLABEL option of COPYFILE command 61
 tracing
 EXEC 2 programs 550
 resuming after temporary halting 666
 start, for System Product Interpreter or EXEC 2
 exec 667
 suspending for System Product Interpreter or EXEC
 2 exec 667
 suspending recording temporarily 667
 SVC instructions 602
 halting 665
 System Product Interpreter programs 550
 trailing fill characters, removing from records 62
 TRANS option of COPYFILE command 63
 TRANSFER BUFF option of TAPE command 614
 TRANSFER IMMED option of TAPE command 614
 transient area
 CMS commands that execute in 10
 creating modules to execute in 230
 transient directories in VSE, displaying 135
 TRANSLATE CHARACTER (.TR) format word 810
 TRANSLATE option
 of CMS QUERY command 449
 of CMS SET command 586
 translate tables
 defining input characters for translation 560
 translate tables (*continued*)
 defining output characters for translation 560
 displaying 435
 displaying system translate tables 449
 displaying user translate tables 449
 setting system translations 586
 setting user translations 586
 translation list for COPYFILE command,
 description 63
 TRC option of PRINT command 404
 TRTCH
 of ASSGN command 44
 of FILEDEF command 193
 of TAPE command 614
 TRUNC
 of commands 6
 option of COPYFILE command 62
 truncation
 of command names
 querying acceptability of 417
 of records in CMS file 62
 of trailing blanks from CMS file 62
 TS (Trace Start) immediate command 667
 two-color ribbon, controlling use of 577
 TXTLIB
 command
 ADD option 629
 DEL option 629
 description 629
 DISK option 630
 FILENAME option 630
 GEN option 629
 MAP option 629
 PRINT option 630
 TERM option 630
 files
 adding members 629
 creating members 629
 deleting members 629
 determining which TXTLIBs are searched 450
 identifying for LOAD and INCLUDE command
 processing 237
 listing members in 629
 maximum number of members 631
 search for unresolved references 316, 317
 searched during INCLUDE command
 processing 271
 searched during LOAD command processing 308
 file, searching for unresolved references 271
 option
 of CMS QUERY command 450
 of GLOBAL command 237
 TYPE
 command
 COL option 633
 description 633
 HEX option 633
 MEMBER option 633

TYPE (continued)

- option
 - of COPYFILE command 61
 - of ERASE command 142
 - of IDENTIFY command 267
 - of INCLUDE command 272
 - of LOAD command 309
 - of NAMEFIND command 350
 - of RENAME command 489
 - of SENDFILE command 529
- types of authority displayed by LISTFILE 297
- types of lock 79

U

- UA option
 - of ASSGN command 44
 - of LISTIO command 300
- UNPACK option of COPYFILE command 62
- unresolved references
 - during MODULE execution 230
 - loader handling of 316
 - resolving with INCLUDE command 273
 - searching for TEXT files 310
 - searching TXTLIBs for 310
- UNTIL option of XEDIT command 653
- UPCASE option
 - of ASSGN command 44
 - of COPYFILE command 62
 - of FILEDEF command 193
 - of PRINT command 404
- UPDATE
 - command
 - control statements 638
 - CTL option 637
 - description 636
 - DISK option 637
 - error handling for 646
 - INC option 637
 - input files 642
 - NOCTL option 637
 - NOINC option 637
 - NOREP option 636
 - NOSEQ8 option 637
 - NOSTK option 637
 - NOTERM option 637
 - OUTMODE option 637
 - output files 641
 - PRINT option 637
 - REP option 636
 - SEQ8 option 636
 - STK option 637
 - STOR option 637
 - TERM option 637
 - control statements
 - COMMENT 641
 - DELETE 640
 - INSERT 639
 - REPLACE 640

UPDATE (continued)

- control statements (continued)
 - SEQUENCE 638
- option
 - of TAPPDS command 623
 - of XEDIT command 653
- update log
 - for UPDATE command operations 637
 - generating at your terminal 637
- updating a virtual screen and associated window 719
- updating a virtual screen from an exec 767
- UPDIRT option of RENAME command 489
- uppercase letters
 - converting to lowercase with COPYFILE command 62
- UPSI (user program switch indicator)
 - byte
 - querying setting of 451
 - setting 588
 - option
 - of CMS SET command 588
 - option of CMS QUERY command 451
- user catalogs 125
 - identifying
 - in CMS/DOS 121
- user file directory
 - contents 27
 - creating 27
 - updating on disk 481
- USER option
 - of CONVERT COMMANDS command 55
 - of EXECDROP command 153
 - of EXECLOAD command 171
 - of EXECMAP command 173
- user program area
 - commands that execute in 10
- user program switch indicator (UPSI)
 - byte
 - querying setting of 451
 - setting 588
 - option
 - of CMS SET command 588
 - option of CMS QUERY command 451
- user-defined synonyms, displaying 448
- user-written commands
 - assigning synonyms for 606
 - creating 226

V

- VALIDATE command
 - description 649
 - examples 649
 - format 649
 - messages 650
 - operands
 - fn ft fm 649
 - * (asterisk) 649

VALIDATE command (*continued*)

- responses 650
- usage notes 649
- VAR option of EXECIO command 160
- variable length
 - converting to fixed-length 62
- variable size window, displaying 697
- variable symbols
- verifying a VSAM catalog structure 47
- verifying that a disk or directory is accessed 649
- verifying the syntax of file identifier 649
- virtual disks
 - See also* disks
 - counting cylinders on 220
 - initializing 220
 - resetting number of cylinders on 220
 - valid addresses for 220
- virtual machine
 - components of 1
 - definition 1
 - environment, determining status of 415, 704
- virtual screen
 - changing attributes 757
 - clearing 674
 - creating 679
 - data area 676
 - default connections to windows 744
 - default virtual screens 743
 - defining 679
 - defining fields 772
 - deleting 686
 - displaying information about 714
 - displaying location of cursor 708
 - entering information 772
 - field definition character 768
 - naming 681
 - numbering of reserved lines 677
 - positioning the cursor 676
 - querying 714
 - reserved area 676
 - specifying character attributes 772
 - suppressing updates 758
 - updating 719
 - updating from an exec 767
 - updating the associated plane buffers 772
 - updating with data 770
 - using character attributes when displaying data 732
 - writing data 772
 - writing data to a CMS file 702
 - writing lines from a file to a virtual screen 690
 - XEDIT virtual screen 652
- virtual screen, CMS commands for
 - CLEAR VSCREEN 674
 - CURSOR VSCREEN 676
 - DEFINE VSCREEN 679
 - DELETE VSCREEN 686
 - GET VSCREEN 690
 - PUT SCREEN 700

virtual screen, CMS commands for (*continued*)

- PUT VSCREEN 702
- SET VSCREEN 757
- WAITREAD VSCREEN 767
- WAITT VSCREEN 770
- WRITE VSCREEN 772
- Virtual Storage Access Method (VSAM)
 - catalogs
 - determining which catalog is searched 125
 - identifying 125
 - identifying in CMS/DOS 121
 - verifying a structure of 47
 - data set extents displayed 119
 - determining free space extents 285
 - environment, resetting 176
 - files
 - defining with DLBL command 114
 - specifying extents 116
 - master catalog
 - identifying 125
 - identifying in CMS/DOS 121
 - option
 - of DLBL command 116
 - of SET DOS ON command 545
 - restriction
 - for OS/VS users 815
 - for VSE users 815
 - OS/VS users 817
- VM/SP (Virtual Machine/System Product) basic description 1
- VOLID operand
 - of FILEDEF command 198
 - of LABDEF command 278
- VOLSEQ operand of LABELDEF command 278
- VSAM (Virtual Storage Access Method)
 - catalogs
 - determining which catalog is searched 125
 - identifying 125
 - identifying in CMS/DOS 121
 - verifying a structure of 47
 - data set extents displayed 119
 - determining free space extents 285
 - environment, resetting 176
 - files
 - defining with DLBL command 114
 - specifying extents 116
 - master catalog
 - identifying 125
 - identifying in CMS/DOS 121
 - option
 - of DLBL command 116
 - of SET DOS ON command 545
 - restriction
 - for OS/VS users 815
 - for VSE users 815
 - OS/VS users 817
- VSCREEN option
 - of CMS QUERY command 714

VSCREEN option (*continued*)
of CMS SET command 757
VSE/VSAM Catalog Check Service Aid, invoking 47

W

WAITD Macro
See VM/SP Application Development Reference for CMS

WAITECB Macro
See VM/SP Application Development Reference for CMS

WAITRD Function
See VM/SP Application Development Reference for CMS

WAITREAD VSCREEN command 671
description 767
execution described 767
format 767
messages 769
operands
vname 767
usage notes 767
using with WRITE VSCREEN 767

WAITT Macro
See VM/SP Application Development Reference for CMS

WAITT VSCREEN command 671
description 770
format 770
messages 770
operands
vname 770
* (asterisk) 770
usage notes 770

WIDTH option of XEDIT command 652

window
changing attributes 759
changing location 699
changing number of lines and columns 766
clearing 675
connecting to a virtual screen 764
creating 683
default connections to virtual screens 744
default windows 741
defining 683
defining borders 729
deleting 687
displaying a variable size window 764
displaying information about 715
displaying location indicator 749
displaying number of reserved lines 712
dropping 688
dropping the WM window 688
expanding the size to the physical screen size 694
hidden windows, displaying information about 709
hiding 692
placing at top of display order 764

window (*continued*)
popping 697
positioning 699
reducing a window size to one line 696
restoring after maximizing or minimizing 720
scrolling the topmost window 698
specifying reserved lines 754
XEDIT window 652

WINDOW option
of CMS QUERY command 715
of CMS SET command 759
querying 715

WINDOW option of XEDIT command 652

windowing commands 669

windows, CMS commands for

ALARM VSCREEN 673

B (backward) 779

C (clear) 779

CLEAR WINDOW 675

D (drop) 779

DEFINE WINDOW 683

DELETE WINDOW 687

DROP WINDOW 688

F (forward) 780

full-screen CMS commands 669

H (hide) 780

HIDE WINDOW 692

L (left) 780

M (move) 781

MAXIMIZE WINDOW 694

MINIMIZE WINDOW 696

N (minimize) 781

O (restore) 781

P (pop) 782

POP WINDOW 697

POSITION WINDOW 699

R (right) 782

REFRESH 719

RESTORE WINDOW 720

ROUTE 721

S (size) 782

SCROLL 724

SET BORDER 729

SET CMSPF 734

SET FULLSCREEN 739

SET LOCATION 749

SET LOGFILE 750

SET WINDOW 759

SET WMPF 761

SHOW WINDOW 764

SIZE WINDOW 766

X (maximize) 783

WM window

automatically displayed 698, 741

CLEAR key 745

commands you can enter 697

displaying 697

dropping 688

WM window (*continued*)
 exiting 688
 PA2 key 745

WMPF keys
 canceling 762
 changing settings 761
 displaying definitions 716
 RETRIEVE function 762
 using NOWRITE option 762

WMPF option
 of CMS QUERY command 716
 of CMS SET command 761

WRITE VSCREEN command 671
 description 772
 examples 775
 format 772
 messages 776
 operands
 BLANKS 773
 col 772
 color 773, 774
 DATA 773
 exthi 773, 774
 FIELD 773
 HIGH 773
 INVISIBLE 773
 length 772
 line 772
 NOHIGH 773
 NOPROTECT 773
 NULLS 773
 PROTECT 773
 PSS 774
 psset 773
 RESERVED 772
 text 774
 vname 772
 usage notes 774
 writing lines from a file to a virtual screen 690
 writing virtual screen data to a CMS file 702

WRTAPE Macro
See VM/SP Application Development Reference for CMS

WRTERM Macro
See VM/SP Application Development Reference for CMS

WTM
 option of TAPE command 612
 tape control function 611

WVOL1 operand of TAPE command 612

X

X border command 783
 XCAL option of LKED command 303
 XEDIT command 651
 CTL option 653
 editing a MACLIB member 652

XEDIT command (*continued*)
 INCR option 654
 MEMBER option 652
 MERGE option 653
 NOCLEAR option 652
 NOCTL option 653
 NOMSG option 652
 NOPROFIL option 652
 NOSCREEN option 652
 NOSEQ8 option 653
 NOUPDATE option 653
 PROFILE option 652
 SEQ8 option 653
 SIDCODE option 654
 UNTIL option 653
 UPDATE option 653
 usage 654
 WIDTH option 652
 WINDOW option 652

XEDIT option
 of LISTDIR command 282
 of LISTFILE command 294
 of NAMEFIND command 350

XMITMSG command
 description 658
 examples 661
 format 658
 messages 663
 operands
 APPLID 659
 CALLER 659
 COMPRESS 659
 DISPLAY 660
 ERRMSG 660
 FORMAT 658
 HEADER 660
 LETTER 659
 LINE 659
 msgnumber 658
 nn 659
 NOCOMPRESS 659
 NODISPLAY 660
 NOHEADER 660
 sublist 658
 SYSLANG 660
 VAR 659
 * (asterisk) 659
 usage notes 661

XREF option
 of ASSEMBLE command 38
 of LKED command 303
 of OPTION command 393

XTENT option of FILEDEF command 192

Y

Y-disk accessed after IPLing CMS 29

YFLAG option of ASSEMBLE command 40

Numerics

18-track tapes, specifying on TAPE command 613
18TRACK option of FILEDEF command 193
19E virtual disk address accessed as file mode Y 29
190 virtual disk address accessed as file mode S 29
191 virtual minidisk address accessed as file mode
A 29
192 virtual disk address accessed as file mode D 29
195 virtual disk address formatted by CMS batch
facility 50
3480 Magnetic Tape Subsystem used with TAPE
command 614
3800 printer, loading a virtual, via SETPRT
command 590
48C option of OPTION command 393
512-byte block formatted disk using with RESERVE
command 493
60C option of OPTION command 393
7-track tapes, specifying on TAPE command 613
7TRACK option
of ASSGN command 44
of FILEDEF command 193
of TAPE command 613
9-track tapes, specifying on TAPE command 613
9TRACK option
of ASSGN command 44
of FILEDEF command 193
of TAPE command 613

Special Characters

.BX (BOX) format word 798
.CM (COMMENT) format word 800
.CS (CONDITIONAL SECTION) format word 801
.FO (FORMAT MODE) format word 803
.IL (INDENT LINE) format word 805
.IN (INDENT) format word 806
.MT (MENU TYPE) format word 807
.OF (OFFSET) format word 808
.SP (SPACE LINES) format word 809
.TR (TRANSLATE CHARACTER) format word 810
./ D (DELETE) UPDATE control statement 640
./ I (INSERT) UPDATE control statement 639
./ R (REPLACE) UPDATE control statement 640
./ S (SEQUENCE) UPDATE control statement 638
./ * (COMMENT) UPDATE control statement 641
\$LISTIO EXEC file
appending information to 300
creating 300
format 301
* (asterisk)
entered in file ID 7
in ACCESS command 28
in COPYFILE command 60
in DLBL command 114

* (asterisk) (*continued*)
in DSERV command 135
in EDIT command 138
in ERASE command 142
in EXECDROP command 153
in EXECIO command 157
in EXECLOAD command 171
in EXECMAP command 173
in EXECSTAT command 178
in FILEDEF command 187
in FILELIST command 203
in FINIS command 219
in INCLUDE command 272
in LABELDEF command 277
in LISTDS command 285
in LISTFILE command 291
in LOAD command 309
in NAMEFIND command 349
in NUCXDROP command 384
in PEEK command 398
in PRINT command 403
in PUNCH command 412
in READCARD command 466
in RENAME command 488
in START command 597
in STATE and STATEW commands 600
in SYNONYM command 606
in TAPE command 610
in TAPPDS command 622
in XEDIT command 651
with DISK option, of CMS QUERY command 427
with RESET option
of INCLUDE command 272
of LOAD command 309
with ROUTE option, of CMS QUERY
command 713
*COPY statement 332
/ (diagonal)
used in ACCESS command 27
used in EXECUTE command 207, 462
? (question mark)
used with DISK option of FILEDEF command 197
used with DSN option of DLBL command 115
= (equal sign)
in COPYFILE command 60, 64
in RDR command 454
in RENAME command 488



Program Number
5664-167

File Number
S370/4300-39

VM/SP
CMS Command Reference
Order No. SC19-6209-05

**READER'S
COMMENT
FORM**

Is there anything you especially like or dislike about this book? Feel free to comment on specific errors or omissions, accuracy, organization, or completeness of this book.

If you use this form to comment on the online HELP facility, please copy the top line of the HELP screen.

_____ **Help Information** line ____ of ____

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you, and all such information will be considered nonconfidential.

Note: Do not use this form to report system problems or to request copies of publications. Instead, contact your IBM representative or the IBM branch office serving you.

Would you like a reply? ___YES ___NO

Please print your name, company name, and address:

IBM Branch Office serving you:

Thank you for your cooperation. You can either mail this form directly to us or give this form to an IBM representative who will forward it to us.

Reader's Comment Form

CUT
OR
FOLD
ALONG
LINE

Fold and tape

Please Do Not Staple

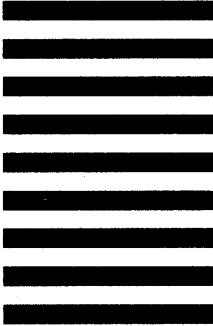
Fold and tape



BUSINESS REPLY MAIL
FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NY

POSTAGE WILL BE PAID BY ADDRESSEE:

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



INTERNATIONAL BUSINESS MACHINES CORPORATION
DEPARTMENT G60
PO BOX 6
ENDICOTT NY 13760-9987



Fold and tape

Please Do Not Staple

Fold and tape





Program Number
5664-167

File Number
S370/4300-39

SC19-6209-05

